Автономная некоммерческая организация высшего образования
**«Открытый гуманитарно-экономический университет»**
**(АНО ВО ОГЭУ)**

**ОСНОВНАЯ ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА**
**(ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА)**
**ВЫСШЕГО ОБРАЗОВАНИЯ - ПРОГРАММА БАКАЛАВРИАТА**

НАПРАВЛЕНИЕ ПОДГОТОВКИ
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**(уровень бакалавриата)**

**Направленность (профиль): Научно-исследовательская деятельность**

**Квалификация - бакалавр**

**ПРИЛОЖЕНИЕ 4**

# МЕТОДИЧЕСКИЕ ПОСОБИЯ И УКАЗАНИЯ ПО ИЗУЧЕНИЮ ДИСЦИПЛИН

Москва
2018

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**КУРСОВАЯ РАБОТА**

ПОРЯДОК НАПИСАНИЯ, ОФОРМЛЕНИЯ И СДАЧИ

**МОСКВА 2018**

Разработано В.Н. Фокиной, к.соц.н., доц.,
М.В. Вольфман, к.п.д.;
Под ред. Н.С. Сельской, к.т.н., проф.

Рекомендовано Учебно-методическим
советом в качестве учебного пособия
для обучающихся

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

# КУРСОВАЯ РАБОТА

ПОРЯДОК НАПИСАНИЯ, ОФОРМЛЕНИЯ И СДАЧИ

Данные методические указания разработаны с целью обеспечения качества подготовки обучающихся в соответствии с требованиями ФГОС ВО.

В методических указаниях подробно рассматриваются методические аспекты подготовки и оформления курсовых работ. Особое внимание уделяется выбору темы курсовой работы, с учетом фактора преемственности разрабатываемой проблемы в последующих курсовых работах и выпускной квалификационной работе, а также отражены наиболее актуальные вопросы самостоятельной работы обучающихся по теме курсовой работы.

Данные указания предназначены для обучающихся, организаторов учебного процесса.

**О Г Л А В Л Е Н И Е**

## 1 ОБЩИЕ ПОЛОЖЕНИЯ

В образовательной организации оценка качества освоения образовательных программ проводится путем осуществления текущего контроля успеваемости, промежуточной аттестации обучающихся и итоговой аттестации выпускников.

Курсовая работа является одним из видов промежуточной аттестации и осуществляется по дисциплинам в соответствии с учебным планом соответствующего направления подготовки. В данном учебно-методическом пособии рассматриваются правила написания курсовых работ.

Курсовая работа - самостоятельная разработка конкретной темы по изучаемой дисциплине с элементами научного анализа, отражающая приобретенные обучающимся теоретические знания и практические навыки, умение работать с литературой, анализировать источники, делать обстоятельные и обоснованные выводы. Указанные элементы подготовки курсовой работы позволяют ее отнести к разряду творческих работ.

Письменные работы по дисциплинам учебного плана являются важным этапом в освоении основной образовательной программы обучающимися, способствующим как приобретению навыков самостоятельного научного и практического подхода к освоению учебного материала, так и формированию профессиональных компетенций. Кроме того, письменные курсовые работы позволяют эффективно осуществлять контроль за самостоятельной работой обучающихся и оценивать, наряду с экзаменами и зачетами, подготовленность будущего специалиста.

Курсовые работы выполняются по окончании изучения дисциплин, определенных учебными планами по каждому направлению подготовки. В учебном плане указывается наименование дисциплины, по которой запланировано выполнение курсовой работы, семестр и вид отчетности (дифференцированный зачет).

Курсовые работы являются обязательными этапами, предшествующими написанию и защите выпускной квалификационной работы.

Обучающемуся необходимо помнить, что он лично отвечает за качество и оформление курсовой работы.

## 2 ОРГАНИЗАЦИЯ КОНСУЛЬТАТИВНОЙ РАБОТЫ

Реализация образовательной организацией электронного обучения и дистанционных образовательных технологий с использованием компьютерных сетей дает возможность обучающимся образовательной организации получать профессиональные консультации высококвалифицированных специалистов через систему IP-хелпинг и телетьюторинги, тем самым обеспечивая требуемое качество обучения в любом центре доступа, независимо от его местонахождения.

Опыт руководства выпускными квалификационными работами (ВКР) в образовательной организации показал, что, начиная уже с первого курса, обучающемуся необходимо ориентироваться на подготовку будущей выпускной квалификационной работы: научиться работать с литературой, пользоваться методическими материалами при написании и оформлении письменных творческих работ (рефератов, статей, эссе, курсовых работ).

При написании курсовых работ обучающиеся могут использовать материалы:

- слайд-лекции;
- телетьюторинги по дисциплине (модулю);
- издания образовательной организацией в печатном или электронном виде (методические пособия по изучению дисциплины (модулю), методические пособия по курсовым работам по дисциплине, данные методические указания, методические указания (1498.01.01;МУ.01;4 «Введение в технологию обучения. Основные правила оформления учебно-научных и творческих работ»);
- материалы, размещены в телекоммуникационной двухуровневой библиотеке, в том числе в «Виртуальном читальном зале» ТКДБ.

Все это позволяет обучающемуся в индивидуальном режиме активно вести поиск ответов на возникающие вопросы по выбору темы, поиску литературы и пр.

Для подготовки курсовой работы в образовательной организации отводится шесть академических часов, которые распределены следующим образом:

- просмотр телетьюторингов по конкретным дисциплинам;
- работа с текстами в ТКДБ;
- консультации «вопрос-ответ» в системе IP-Хелпинг.

Руководство курсовыми работами осуществляет профессорско-преподавательский состав (ППС) образовательной организации через систему IP-Хелпинг. Консультирование в системе IP-Хелпинг, отвечает на различные вопросы, в том числе:

- формирование структуры курсовой работы (соответствие выбранной теме, самостоятельно разработанному рабочему плану КР);
- оказание помощи в формировании списка литературы;
- конкретные вопросы содержательного характера (правильность определения целей, задач, методов исследования, содержания приложений и т.п.).

Порядок работы в системе IP-Хелпинг регулируется руководством пользователя, действующим в образовательной организации * («Открытая Автоматизированная информациионная система (ОАЗИС). «Система IP-Хелпинг». Руководство пользователя»).

Ответы ведущих преподавателей и специалистов образовательной организации в системе IP-Хелпинг на вопросы содержательного и научного характера размещаются, как правило, в течение 3-4 дней.

Таким образом, качество курсовой работы зависит не только от уровня знаний обучающегося, но и от его активности и профессионализма в использовании современных информационно-технических средств (Internet, IP-Хелпинг, телетьюторинги и т.д.), предоставляемых образовательной организацией. При этом объем материалов дает возможность обучающемуся подготовить курсовую работу не только на требуемом уровне, но и в соответствии с современным уровнем развития науки и техники.

## 3 ТРЕБОВАНИЯ К СОДЕРЖАНИЮ КУРСОВОЙ РАБОТЫ

### 3.1 Этапы выполнения курсовой работы

В предлагаемых рекомендациях по подготовке курсовых работ отражены наиболее актуальные вопросы самостоятельной работы обучающихся по теме курсовой работы, от решения которых во многом зависит качество выполнения работы и соблюдение установленных сроков ее представления.

Процесс выполнения курсовой работы включает в себя ряд взаимосвязанных этапов, типовой перечень которых (в порядке выполнения) представлен ниже:

– выбор темы проводится с учетом преемственности разрабатываемой темы в последующих курсовых работах и выпускной квалификационной работе;

– формирование структуры курсовой работы и графика ее выполнения;

– сбор, анализ и обобщение изученного материала по выбранной теме;

– формулирование основных теоретических положений, практических выводов и рекомендаций по результатам анализа;

– оформление курсовой работы, списка использованных источников и литературы, глоссария и приложений;

* Прежде чем задать свой вопрос, обучающийся должен просмотреть перечень вопросов, сформулированных ранее другими обучающимися, и найти искомый ответ в системе «ОАЗИС» или в телекоммуникационной двухуровневой библиотеке (Фонд «IP-Хелпинг»).

– проверка чистового варианта курсовой работы, с помощью ПО Нормоконтроль на сайте «Личная студия» (http://edu.muh.ru), устранение выявленных недостатков;

– размещение электронного варианта курсовой работы в шаблон через сайт «Личная студия» для оценки ее качества интеллектуальным роботом контроля оригинальности и профессионализма (ИР КОП).

Файл шаблона «Курсовая работа» используется для формирования курсовой работы, её транспортировки в базовый вуз и последующего хранения. Порядок размещения курсовой работы в электронный шаблон регламентируется «Технологической инструкцией по заполнению электронного шаблона курсовой работы / курсового проекта и передаче его в базовый вуз».

Неудовлетворительная оценка или отсутствие аттестации из-за невыполнения обучающимся курсовой работы расцениваются как академическая задолженность, ликвидация которой осуществляется в установленном порядке.

### 3.2 Выбор темы

Тематика курсовых работ определяется централизованно базовым вузом. Выбор темы работы осуществляется обучающимся, исходя из уровня понимания и осознания актуальности темы, оценки ее теоретического и практического значения. Для своевременного ознакомления с темами курсовых работ, утвержденный список размещен в телекоммуникационной двухуровневой библиотеке (ТКДБ), а также приводится в методических пособиях по курсовым работам соответствующих направлений подготовки.

Обучающийся может выбрать тему работы из предлагаемого перечня тем курсовых работ либо, исходя из собственных научных и практических интересов, предложить свою тему для исследования, не выходя за рамки изучаемой дисциплины. Темы должны отвечать современным требованиям развития науки, экономики, культуры и образования. Кроме того, необходимость разработки именно этой темы должна быть обоснована и в установленном порядке утверждена базовым вузом.

Свобода выбора тем курсовых работ позволяет реализовать индивидуальные научные интересы обучающегося, своеобразие подхода к изучению проблемы.

Уже на 1-м курсе (на вводной лекции) обучающиеся знакомятся со структурой учебного плана по направлению подготовки, где определены дисциплины, по которым предусмотрено написание курсовых работ. Идеи для выбора темы могут возникнуть из докладов и выступлений, при просмотре проблемных лекций, по результатам проведения практик и т.д.

Опыт образовательной организации показал, что наибольшую трудность вызывает выполнение первой курсовой работы. Поэтому на младших курсах рекомендуется выбирать более узкие, конкретные темы, так как это дает возможность глубже вникнуть в проблематику курсовой работы, избежать поверхностности, описательного характера излагаемого материала. Соразмерность задачи обеспечит обучающемуся интерес к выполняемой работе.

Тема должна раскрываться таким образом, чтобы она приближалась по своей направленности к небольшому исследованию и заключала постановку проблемы, указание задач, аргументацию, анализ материала, примеры, выводы.

Формулирование темы должно:

- соответствовать содержанию, ограничивать круг вопросов, которые разрабатывает автор, раскрывать исследуемую идею;

- отражать замысел автора, т.е. полное или частичное изложение проблемы;

- быть достаточно развернутым, чтобы показать рамки исследования, но при этом не содержать лишних слов.

Особое внимание следует уделить выбору темы курсовой работы, с учетом фактора преемственности разрабатываемой проблемы в последующих курсовых работах и выпускной квалификационной работе. Структура работы при этом остается неизменной, но углубляется фундаментальность исследования проблемы,

привлекается большее количество нормативных, правовых документов, расширяется список источников литературы по теме и пр.

### 3.3 Структура курсовой работы. Разработка рабочего плана

Для разработки рабочего плана курсовой работы, обучающийся должен четко представлять ее структуру. Поскольку структура курсовой работы, независимо от дисциплины и темы, остается неизменной и сходна со структурой выпускной квалификационной работы (ВКР), в основе которой могут лежать материалы курсовых работ, которые были выполнены обучающимся за время обучения в образовательной организации.

Содержательная часть курсовой работы имеет следующую структуру: введение; основную часть; заключение; список использованных источников; глоссарий, приложения.

Унифицированные требования, предъявляемые в образовательной организации к объему и оформлению курсовой работы, приведены в приложении А; с подробным изложением требований обучающийся должен знакомится в методических указаниях (1498.01.01;МУ.01;5 «Введение в технологию обучения. Основные правила оформления учебно-научных и творческих работ»).

Для рациональной организации самостоятельной работы в ходе выполнения курсовой работы обучающемуся необходимо разработать план, который позволит более продуктивно организовать исследовательскую работу по избранной теме. Рабочий план составляется в произвольной форме, в котором отражаются конкретные этапы по написанию курсовой работы и сроки их реализации в соответствии с учебным планом.

Примерный план подготовки курсовой работы приведен в приложении Б.

План составляется обучающимся на основе предварительного ознакомления с литературой и другими источниками. Приступая к работе над планом, первоначально необходимо проанализировать вопросы темы по учебникам, в частности по рабочим учебникам, так как в них материалы изложены в сжатом виде. После этого целесообразно переходить к специальной литературе, изучению первоисточников, нормативных документов. Такой способ - от простого к сложному - обеспечит возможность «не потеряться» в обилии фактов, идей, авторов.

### 3.4 Сбор, анализ и обобщение материала

С выбором темы неразрывно связаны подбор и изучение обучающимся литературы.

Этап сбора, анализа и обобщения материала по теме является наиболее трудоемким, сложным и ответственным этапом исследовательской работы, когда определяются степень научной разработанности проблемы и содержательные границы курсового исследования.

Обзор литературы по теме исследования (нормативной, первоисточников, научной и учебной) начинается с подготовки списка используемых источников, который должен всесторонне охватывать исследуемую тему.

Источниками для формирования такого списка могут быть:

- список обязательной и рекомендованной литературы в программе учебной дисциплины;
- электронные образовательные ресурсы в сети Internet;
- библиографические списки и сноски в учебниках и научных изданиях (монографиях, научных статьях) последних лет или диссертациях по данной тематике;
- рекомендации преподавателя (IP-хелпинг);
- каталоги телекоммуникационной двухуровневой библиотеки образовательной организации и библиотек, к которым ТКДБ предоставляет доступ в режиме виртуального читального зала.

В первую очередь следует подбирать литературу за последние 3-5 лет, поскольку в ней отражены последние научные достижения по данной проблеме, современное законодательство и практическая деятельность. Использование литературных и иных источников 10-и, 20-и или даже 30-летней давности должно быть скорректировано применительно к современным концепциям ученых и специалистов.

Указание на литературные источники по исследуемой теме можно встретить в сносках и списке литературы уже изданных работ. Поиск статей в научных журналах следует осуществлять просмотром последнего номера соответствующего журнала за определенный год, так как в нем, как правило, помещается указатель всех статей, опубликованных в данном журнале за год.

Следует просматривать профессиональные и специализированные периодические издания (журналы, газеты, сборники научных трудов).

Для подготовки курсовой работы каждый обучающийся образовательной организации имеет уникальную возможность работать с литературой по теме, используя ТКДБ образовательной организации. При этом не имеет значение местонахождение обучающегося, так как доступ к ее ресурсам имеется в любом центре доступа. ТКДБ предоставляет доступ в режиме виртуального читального зала к ресурсам удаленного доступа электронных библиотек:

- Научной электронной библиотеки (НЭБ);
- Открытой русской электронной библиотеки;
- Единого окна доступа к образовательным ресурсам;
- Электронной библиотеки международных документов по правам человека;
- Базы электронных диссертаций "Proquest digital dissertations";
- Портала «Theses Canada» («Канадские полнотекстовые диссертации»);
- Базы журналов открытого доступа «Directory of open access journals» и др.

Для написания учебно-научной работы большой интерес представляет «Единое окно доступа к образовательным ресурсам». В электронной библиотеке «Единого окна» размещены образовательные информационные ресурсы, разработанные ведущими российскими вузами: учебники, тексты лекций, методические указания и другие ресурсы.

В образовательной организации ведется активная научная работа по различным областям знаний (социологии, экономике, менеджменту, психологии, юриспруденции и др.) результаты которой публикуются в разделе «Труды СГУ», который размещен в фонде научной литературы. Фонд научной литературы содержит также обширную подборку монографий и диссертаций по всем научным направлениям образовательной организации.

В ТКДБ представлен широкий круг научных журналов на русском языке по всем областям знаний. Пользователь имеет доступ к алфавитному перечню заглавий журналов и возможность отбора по году выпуска журнала. Также имеются биографические справочники и словари.

В образовательной организации оформлена подписка на коллекцию журналов «Научной электронной библиотеке» по социальным и гуманитарным наукам. Коллекция содержит журналы по социологии, психологии, юриспруденции, образованию, менеджменту и др. Преодолеть языковой барьер поможет система компьютерного перевода в Google.

Работа с научной книгой начинается с изучения титульного листа, где приводятся данные об авторе и выходные сведения (год и место издания), а также оглавления. Год издания книги позволяет соотнести информацию, содержащуюся в ней, с существующими знаниями по данной проблеме на современном этапе. В оглавлении книги раскрываются ключевые моменты ее содержания, логика и последовательность изложения материала.

Далее необходимо познакомиться с введением, где, как правило, формулируется актуальность темы, кратко излагается содержание книги и ее направленность, раскрываются источники и способы исследования, степень разработанности проблемы.

Ознакомление можно завершить постраничным просмотром, обратив внимание на научный аппарат, частично расположенный в сносках, на определения ключевых понятий, полноту изложения заявленных в оглавлении вопросов.

При изучении специальной (научной) литературы необходимо обращаться к различным словарям, энциклопедиям и справочникам в целях выяснения смысла специальных понятий и терминов, конспектируя те из них, которые в дальнейшем будут используемы в тексте работы и при составлении глоссария. Фонд

справочных, нормативных и официальных изданий ТКДБ содержит энциклопедии (отраслевые и универсальные); словари (отраслевые и универсальные); справочники (отраслевые и универсальные).

Изучение нормативных документов – законов, подзаконных актов, постановлений – является обязательным, так как знание этих документов и умение работать с ними – залог успешной в дальнейшем профессиональной деятельности.

В виртуальном читальном зале ТКДБ обучающимся предоставляется возможность удаленного доступа к информационным ресурсам «Электронной библиотеки международных документов по правам человека», в которой размещается информация о различных межправительственных организациях в области прав человека, о проводимых и планируемых конференциях, сессиях органов по контролю за соблюдением международных договоров в области прав человека, а также оперативная информация о принятых решениях, рассматриваемых докладах и отчетах о соблюдении прав человека.

Образовательная организация, являясь пользователем справочно-информационных систем «Гарант» или «КонсультантПлюс», предоставляет возможность каждому обучающемуся быть в курсе последних изменений в законодательстве и решать возможные проблемы в области правовой информации и бухгалтерской документации. Данные системы являются самыми обширными правовыми базами России, которые содержат не только нормативные правовые акты, составляющие основу российского законодательства, но и уникальный банк консультаций экспертов в области налогообложения, обзоры судебной и арбитражной практики, деловую документацию.

В ходе анализа собранного по теме исследования материала выбирают наиболее обоснованные и аргументированные конспективные записи, выписки, цитаты и систематизируют их по ключевым вопросам исследования. На основе обобщенных данных уточняют структуру курсового исследования, его содержание и объем.

Хотя структура работы первоначально определяется на стадии планирования, в ходе написания могут возникнуть новые идеи и соображения, поэтому не рекомендуется окончательно структурировать работу сразу же после сбора и анализа материалов.

### 3.5 Основные части работы

Каждая структурная часть курсовой работы имеет свое назначение. Оформляя работу, автор должен помнить, что каждая структурная часть (содержание, введение, основная часть, заключение, список использованных источников и т.д.) начинается с новой страницы.

**Содержание** (или оглавление) включает в себя заголовки всех разделов (глав, параграфов и т.д.), содержащихся в работе. Обязательное требование – дословное повторение в заголовках содержания (или оглавления) названий разделов, представленных в тексте, в той же последовательности и соподчиненности.

Объем курсовой работы должен составлять 20-25 страниц в формате Microsoft Word в соответствии с требованиями, изложенными в приложении А.

Во **введении** кратко характеризуется проблема, решению которой посвящена курсовая работа. Проблема - это теоретический или практический вопрос, ответ на который неизвестен, и на который нужно ответить. Именно на разрешение проблемы (противоречия) направлена работа.

Важным при определении проблемы является вопрос об ее актуальности, предполагающий вычленение значимости избранной темы. Обучающийся должен убедительно показать, почему именно эта тема является наиболее значимой для теории и практики. Наиболее эффективной работа обучающегося будет в том случае, если рассмотрение выбранной проблемы будет связано с профилем той области знания, в которой он специализируется.

Степень разработанности проблемы. Краткий обзор литературных источников позволяет автору сделать вывод, что именно данная тема не полностью раскрыта и требует дальнейшей разработки. В данной части необходимо показать недостаточность разработанности выбранной темы исследования в научных

исследованиях на современном этапе развития общества, необходимость изучения проблемы в новых современных социально-экономических, политических и иных условиях и т.д.

Цель и задачи исследования, которые предполагает раскрыть автор в своей работе.

Цель исследования – это мысленное предвосхищение (прогнозирование) результата, определение оптимальных путей решения задач в условиях выбора методов и приемов исследования в процессе подготовки учебно-научной работы обучающимся.

Задачи исследования в курсовой работе определяются поставленной целью и представляют собой конкретные последовательные этапы (пути) решения проблемы исследования по достижению основной цели.

Методы исследования, использованные в процессе выполнения работы и послужившие инструментом в добывании необходимого фактического материала. Метод – это совокупность приемов. Другими словами, прием – это часть метода.

Например, при исследовании можно использовать следующие методы:

- изучение и анализ научной литературы;
- изучение и обобщение отечественной и зарубежной практики;
- моделирование, сравнение, анализ, синтез, интервьюирование и т.д.

Практическая значимость. Практическая значимость заключается в возможности использования результатов исследования в практической деятельности, независимо от того – является данная учебно-научная работа теоретической или практической.

Необходимо отметить важное правило - введение, как и заключение, рекомендуется писать после полного завершения основной части. До того, как будет создана основная часть работы, трудно написать хорошее введение, так как автор еще не вполне овладел материалами по теме.

Объем введения для курсовой работы - 2-3 страницы в формате Microsoft Word в соответствии с требованиями, изложенными в приложении А.

**Основная часть** курсового исследования должна соотноситься с поставленными задачами. В зависимости от того, какие задачи стоят перед автором, основная часть делится на 2-3 главы. Главы основной части должны быть соразмерны друг другу по объему. Деление глав на параграфы необязательно, но возможно, если в этом есть необходимость.

Предварительная структура основной части курсовой работы (главы, параграфы) определяется еще на стадии планирования. Однако в ходе написания могут возникнуть новые идеи и соображения, которые не только изменят и уточнят структуру, но и обогатят содержание работы и увеличат ее объем.

Содержанием основной части курсового исследования является теоретическое осмысление проблемы и изложение эмпирического материала. Последовательность изложения того и другого может быть различной. Все зависит от желания и предпочтения автора.

Чаще всего вначале излагаются основные теоретические положения по исследуемой теме, а затем - конкретный практический материал, который аргументировано, подтверждает изложенную теорию.

Но возможна и другая последовательность, когда вначале анализируется конкретный материал, а затем на основе этого анализа делаются теоретические обобщения и выводы.

В конце каждой главы должны быть сформулированы краткие выводы.

Обязательным атрибутом исследования является краткий обзор привлеченных источников и литературы. Обзор литературы может быть приведен во введении или в основной части исследования, где рассматриваются теоретические аспекты проблемы.

В ряде случаев обзор источников и литературы выделяют в отдельный параграф основной части исследования, при этом разделяют обзор первоисточников и обзор собственно литературы. Под первыми понимают тексты, которые являются объектом исследования. К ним относятся исторические материалы, законодательные и иные нормативные документы. Под вторыми – литературные источники, которые используются, но при этом не являются предметом исследования. Умение различать эти две группы источников чрезвычайно важно.

Объем основной части курсовой работы 15-20 страниц в формате Microsoft Word в соответствии с требованиями, изложенными в приложении А.

**Заключение** содержит краткую формулировку результатов, полученных в ходе работы. В заключении, как правило, автор исследования суммирует результаты осмысления темы, выводы, обобщения и рекомендации, которые вытекают из его работы, подчеркивает их практическую значимость, а также определяет основные направления для дальнейшего исследования в этой области знаний.

Необходимо иметь в виду, что введение и заключение никогда не делятся на части.

Объем заключения примерно равен объему введения.

**Глоссарий** – толковый (объясняющий) словарь понятий и терминов.

В образовательной организации при выполнении всех учебно-научных работ предусмотрено составление глоссария, который является обязательным компонентом такого вида работ.

Используя в тексте курсовой работы специальные термины, уместно применяя и правильно раскрывая их содержание, автор показывает степень включенности в сферу профессии и готовность к профессиональной и научной деятельности.

В глоссарий включаются основные профессиональные термины (а также их английские либо латинские аналоги, в необходимых случаях аналоги на других языках), факты, персоналии, важнейшие даты, используемые в работе. При оценивании учебно-научных работ обучающихся учитывается количественное и качественное наполнение глоссария.

Глоссарий курсовой работы должен содержать не менее 10 основных понятий и терминов, используемых в контексте исследуемой проблемы.

**Список использованных источников** является обязательным атрибутом курсовой работы.

Список должен содержать сведения обо всех источниках, использованных, цитированных или упоминаемых в работе документах.

В списке использованных источников курсовой работы следует привести не менее 10 библиографических описаний документальных и литературных источников.

**Список сокращений**, если он окажется необходимым в курсовой работе, должен включать в себя расшифровку наиболее часто упоминаемых в работе сокращенных наименований документов, научно-исследовательских институтов, предприятий, акционерных обществ, понятий, слов и т.д. В тексте учебно-научных работ следует избегать сокращений слов, за исключением общепринятых. Считается, что чем меньше сокращений слов и словосочетаний употребляется в научной работе, тем грамотнее она оформлена.

**Приложения** являются обязательным компонентом курсовой работы. В приложениях следует приводить различные вспомогательные материалы (таблицы, схемы, раздаточный материал, графики, диаграммы, иллюстрации, копии постановлений, договоров, инструкции, вспомогательные расчеты и т.п.). С одной стороны, они призваны дополнять и иллюстрировать основной текст, с другой, - разгружать его от второстепенной информации. Все материалы, помещенные в приложениях, должны быть связаны с основным текстом, в котором обязательно делаются ссылки на соответствующие приложения.

Приложения не засчитываются в заданный объем работы.

### 3.6 Изложение результатов работы

Основными целями и задачами написания курсовых работ является не только расширение, углубление и контроль знаний обучающихся, но и формирование умения анализировать теоретический и практический материал, логично, последовательно, ясно, кратко и в тоже время, емко излагать свои мысли в письменном виде.

При написании курсовой работы обучающиеся становятся авторами, многие - впервые. Но к авторской работе предъявляются высокие требования, как по содержанию, так и по оформлению.

В соответствии с целями и задачами курсовая работа не должна быть пересказом изученного материала или простой компиляцией (компиляция - несамостоятельное произведение, составленное путем заимствований, без собственных наблюдений и выводов), из фрагментов используемых статей и книг.

Курсовая работа является собственной интерпретацией проблемы, напоминающей школьное сочинение на свободную тему по литературе или публицистическую статью, так как основывается либо на научной проблеме, либо на учебной и опирается на источники и вторичную научную литературу.

Таким образом, курсовая работа должна представлять собой целостную, однородную и завершенную научную работу обучающегося, в которой должны быть четко сформулированы проблема и исследовательские вопросы, обоснована их актуальность, изложены степень изученности проблемы и состояние ее исследования.

При написании текста курсовой работы автору необходимо следить за тем, чтобы в ходе изложения не терялась основная мысль. Она должна быть видна не только специалисту по данной теме, но и читателю, не посвященному в данную проблемную область. Следует постоянно контролировать соответствие содержания главы или параграфа их заголовкам. Если при написании текста мысль отклонилась от темы, ее следует вернуть в нужное «русло», либо скорректировать структуру работы в соответствии с фактическим ходом изложения. Конец каждой главы, параграфа или абзаца должен иметь логический переход к следующему.

Курсовая работа должна быть написана хорошим научным языком, то есть с соблюдением общих норм литературного языка, правил грамматики и с учетом особенностей научной речи - точности и однозначности, терминологии и стиля.

В современной научной литературе личная манера изложения уступила место безличной. Не употребляются личные местоимения «я» и «мы». Например, вместо фразы «я предполагаю...» можно сказать «предполагается, что...» и т.д.

### 3.7 Оформление работы

Этап оформления курсовой работы является не менее важным, чем остальные, так как на этом этапе автор должен не только свести все материалы по работе в единый документ, но и оформить ее в соответствии с требованиями. Правила, регламентирующие оформление учебно-научных и творческих работ, а также оформление научно-справочного аппарата к ним (цитаты, ссылки, сноски, список использованных источников), изложены в методических указаниях «Введение в технологию обучения. Основные правила оформления учебно-научных и творческих работ» (1498.01.01;МУ.01;5) обязательны для всех обучающихся.

При оформлении глоссария автор проверяет соответствие понятий, данных в тексте, с понятиями, приведенными в глоссарии. Количество понятий, приведенных в глоссарии, должно полностью соответствовать количеству понятий, используемых в тексте. Следует приводить четкие определения понятий, терминов, а не пояснения к ним.

Нельзя включать в глоссарий понятия, выраженные несколькими различными терминами, например, «сырье и основные материалы». Комментарий должен быть конкретным, научным и достоверным. Глоссарий составляется по алфавиту в табличной форме, предусматривающей три графы (столбца). Лексические единицы в глоссарии систематизируются в алфавитном порядке. Образец оформления глоссария представлен в приложении В.

К оформлению чистового варианта курсовой работы приступают после внесения собственных дополнений и изменений.

Обязательными структурными элементами электронного шаблона «Курсовой работы» являются (приложение Г):

- основные сведения о работе;
- содержание;
- введение;
- основная часть;
- заключение;

- глоссарий;
- список использованных источников;
- список сокращений;
- приложения.

Каждый структурный элемент курсовой работы должен начинаться с новой страницы.

Все перечисленные структурные элементы являются обязательными, кроме элемента «Список сокращений» и главы 3 раздела «Основная часть».

После подготовки чистового варианта необходимо еще раз отредактировать текст, устранить опечатки. Далее следует проверить логику работы - насколько точен смысл абзацев и отдельных предложений, соответствует ли содержание глав их заголовкам.

Затем следует проверить, нет ли в работе пробелов в изложении и аргументации, устранить стилистические погрешности, обязательно проверить точность цитат и ссылок, правильность оформления, обратить внимание на написание числительных и т.д. Лишь после такой корректуры окончательный вариант работы следует проверить на соответствие унифицированным требованиям к оформлению курсовых работ (приложение А) с помощью ПО Нормоконтроль.

Целенаправленная завершающая работа с текстом характеризует ответственность автора за представляемый материал. Правила оформления учебно-научных работ являются общими для всех направлений и регламентируются действующими федеральными государственными образовательными стандартами. Поэтому их следует запомнить еще при написании первой курсовой работы, что сэкономит много времени и сил в дальнейшем.

## 4 ОЦЕНКА КАЧЕСТВА КУРСОВОЙ РАБОТЫ

### 4.1 Порядок передачи курсовой работы в базовый вуз

В соответствии с реализуемыми в образовательной организации электронным обучением и дистанционными образовательными технологиями и обеспечением контроля за выполнением учебных планов и повышения качества образовательного процесса в образовательной организации создан комплекс автоматизированной проверки творческих работ обучающихся (ИР КОП). Это позволяет увеличить скорость проверки работ, оптимизировать контроль сроков и качества их выполнения. Результаты автоматизированной проверки курсовых работ поступают в электронные досье обучающихся.

Курсовые работы, выполненные обучающимися по месту обучения, для проведения аттестации, размещаются им самостоятельно на сайте «Личная студия» (http://edu.muh.ru). Первоначально курсовая работа проверяется через ПО Нормоконтроль, после получения положительного заключения проводится оценка качества курсовой работы с помощью интеллектуального робота контроля оригинальности и профессионализма (ИР КОП). Правила заполнению электронного шаблона курсовой работы описаны в соответствующей технологической инструкции по заполнению электронного шаблона курсовой работы/курсового проекта и передаче его в базовый вуз.

### 4.2 Критерии оценивания курсовой работы

Качество письменной творческой работы определяется степенью ее соответствия совокупности установленных требований: она должна быть актуальной, соответствовать выбранной теме исследования, логично построенной, грамотно изложенной и т.п. Каждое требование представляется в виде документально изложенного критерия - признака, на основе которого производится оценивание творческой работы на соответствие данному требованию. Ввиду многообразия требований к творческой работе оценивание ее качества основывается на принципе многокритериальности. (Положение о порядке оценивания творческих работ обучающихся интеллектуальным роботом)

Для оценивания курсовых работ приняты следующие критерии:

– нормоконтроль (оформление, объем, библиография и др.);

– оригинальность (определение уровня самостоятельности обучающегося при выполнении работы;

– профессионализм (оценивание содержания курсовой работы на соответствие заявленной теме и в какой мере отражены профессиональные термины и понятия по теме исследования);

– соответствие работы нормам современного русского языка (соответствие работы нормам орфографической, пунктуационной, синтаксической и стилистической грамотности);

– актуальность содержания (наличие актуальных нормативно-правовых актов, актуальность фактологического материала);

– общий культурный уровень (использование слов из словаря «Достаточный уровень культуры» по отношению к количеству в тексте работы обучающегося).

Аттестация курсовых работ осуществляется по 4-балльной шкале (отлично, хорошо, удовлетворительно, неудовлетворительно).

### 4.3 Размещение результатов курсовой работы

По результатам выполнения курсовых работ формируется Аттестационный лист (приложение Д). В базовом вузе итоги промежуточной аттестации, в том числе по выполненным курсовым работам, вносятся в электронное академическое досье обучающегося в базе данных ИС «Луч-студент».

В ИС «Луч-студент» автоматически формируется ведомость промежуточной аттестации обучающегося, что позволяет базовому вузу контролировать прохождение учебного плана обучающимися.

При положительных результатах текущего контроля успеваемости и промежуточной аттестации в соответствии с учебным планом, по итогам учебного года в базовом вузе оформляется приказ о переводе обучающегося на следующий курс.

**ГЛОССАРИЙ**

| № п/п | Новое понятие | Содержание |
|---|---|---|
| 1 | **IP-Хелпинг** | индивидуальная асинхронная консультация через Интернет, во время которой обучающийся задает вопросы преподавателю по определенной дисциплине, а ведущий преподаватель готовит ответ на специальном сайте образовательной организации |
| 2 | **Академическое досье обучающегося** | совокупность документов, сведений, отражающих кадровые данные, академический статус, успеваемость и др., помещенных в электронную базу ИИС «Луч-студент» |
| 3 | **Базовый вуз** | образовательное учреждение, реализующее полный цикл обучения и осуществляющее организационное, научное и методическое обеспечение учебного процесса в своих территориально структурных подразделениях |
| 4 | **Библиографическое описание** | совокупность библиографических сведений о документе, приведенных по определенным правилам и предназначенных для его идентификации и общей характеристики |
| 5 | **Выпускная квалификационная работа** | завершенная научно-практическая работа выпускника по определенной проблеме, систематизирующая, закрепляющая и расширяющая теоретические знания и практические навыки выпускника при решении конкретной задачи, умение самостоятельно решать профессиональные задачи, характеризующая итоговый уровень его квалификации и подтверждающая его способность к профессиональной деятельности |
| 6 | **Глоссарий** | толковый (объясняющий) словарь понятий и терминов |
| 7 | **Дистанционные образовательные технологии** | образовательные технологии, реализуемые в основном с применением информационных и телекоммуникационных технологий при опосредованном (на расстоянии) или не полностью опосредованном взаимодействии обучающегося и педагогического работника |
| 8 | **Итоговая аттестация** | комплексная оценка уровня подготовки выпускника высшего учебного заведения на соответствие требованиям федерального государственного образовательного стандарта |
| 9 | **Компиляция** | несамостоятельное произведение, составленное путем заимствований, без собственных наблюдений и выводов |
| 10 | **Курсовая работа** | самостоятельная разработка конкретной темы по изучаемой дисциплине с элементами научного анализа, отражающая приобретенные обучающимся теоретические знания и практические навыки, умение работать с литературой, анализировать источники, делать обстоятельные и обоснованные выводы |
| 11 | **Личная студия** | сайт, на котором обучающийся может работать с учебными продуктами по дисциплинам, входящими в его индивидуальный учебный план |
| 12 | **Монография** | научное исследование, посвященное одному вопросу, теме |
| 13 | **Нормоконтроль** | процедура, которая проводится с целью поддержания единообразия в структуре и оформлении курсовых работ и не вмешивается в содержание работ |
| 14 | **Промежуточная аттестация** | аттестация по дисциплинам учебного плана соответствующего направления подготовки в форме экзамена, зачета, курсовой работы |
| 15 | **Рабочий учебник** | учебный продукт, предназначенный для самостоятельного изучения |
| 16 | **Реферат** | краткое точное изложение содержания документа, включающее основные фактические сведения и выводы, без дополнительной интерпретации или критических замечаний автора реферата |
| 17 | **Самостоятельная работа обучающегося** | выполнение различных заданий учебного, исследовательского и самообразовательного характера, средство усвоения системы профессиональных знаний, способ познавательной и профессиональной деятельности; формирование навыков и умений творческой деятельности и профессионального мастерства с применением ИКТ-обучения; текстуальные занятия (работа с текстами) и работа с лекционным материалом |
| 18 | **Текущий контроль** | контроль знаний обучающихся в течение семестра по результатам |

| № п/п | Новое понятие | Содержание |
|---|---|---|
| | **успеваемости** | учебных занятий по модулю (просмотр лекций, выполнение домашних заданий, электронное тестирование, коллективные и компьютерные тренинги, лабораторные работы и т.д.) |
| 19 | **Телекоммуникационная двухуровневая библиотека** | организованное хранилище изданий учебной, учебно-методической, научной и справочной литературы на электронном (цифровом) носителе, предназначенное для быстрого поиска и доступа к конкретному изданию |
| 20 | **Телетьюторинг** | занятие по подготовке обучающихся к экзаменам, написанию курсовой работы, практике в виде индивидуального или коллективного просмотра обучающимися видеозаписей телевизионных консультаций преподавателя |
| 21 | **Учебный план** | перечень учебных дисциплин с указанием объема их изучения, в том числе объема аудиторных занятий, с разбивкой по учебным периодам, с указанием видов аттестации и сроков ее проведения |
| 22 | **Учебно-методический комплекс дисциплины** | совокупность учебных и учебно-методических материалов, предназначенных для обеспечения эффективной работы обучающихся по всем видам образовательной деятельности в соответствии с учебным планом основной образовательной программы и является ее составной частью |
| 23 | **Электронное обучение** | образовательный процесс с применением содержащейся в базах данных и используемой при реализации образовательных программ информации и обеспечивающих ее обработку информационных технологий, технических средств, а также информационно-телекоммуникационных сетей, обеспечивающих передачу по линиям связи указанной информации, взаимодействие участников образовательного процесса |
| 24 | **Эмпирический материал** | материал, основанный на опыте |
| 25 | **Эссе** | прозаическое сочинение небольшого объема и свободной композиции, в котором подчеркнуто индивидуальная позиция автора, сочетается с непринужденным, часто парадоксальным изложением, ориентированным на разговорную речь. Эссе могут иметь философский, историко-биографический, публицистический, литературно-критический, беллетристический и другой характер |

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

### Нормативные правовые акты

1. Об образовании в Российской Федерации [Текст] : Федеральный закон от 29 декабря 2012 г. № 273-ФЗ (ред. от 03.07.2016) // СЗ РФ. – 2012. – № 53 (ч. 1). – Ст. 7598.

2. Положение об итоговой государственной аттестации выпускников высших учебных заведений в Российской Федерации [Текст] : Приказ Минобразования РФ от 25 марта 2003 г. № 1155 // Рос. газета. – 2003.

3. Библиографическая запись. Библиографическое описание. Общие требования и правила составления [Текст] : ГОСТ 7.1 - 2003. – Введ. 2004 - 07 - 01. – М. : Изд-во стандартов, 2004.

4. Отчет о научно-исследовательской работе. Структура и правила оформления [Текст] : ГОСТ 7.32 - 2001. - Введ. 2002 - 07 - 01. – М. : Изд-во стандартов, 2001.

### Учебные издания

1. Карпенко, М. П. Телеобучение [Текст] / М. П. Карпенко. – М. : СГА, 2008.

**ПРИЛОЖЕНИЕ А**

**Унифицированные требования к оформлению курсовых работ обучающихся**

| № п/п | Объект унификации | Параметры унификации |
|---|---|---|
| 1 | Формат листа бумаги | А4 |
| 2 | Размер шрифта | 14 пунктов |
| 3 | Название шрифта | Times New Roman |
| 4 | Междустрочный интервал | Полуторный |
| 5 | Кол-во строк на странице | 28-30 строк<br>(1800 печатных знаков) |
| 6 | Абзац | 1,25 см (5 знаков) |
| 7 | Поля (мм) | Левое, верхнее и нижнее – 20, правое – 10. |
| 8 | Общий объем без приложений | 20-25 страниц машинописного текста |
| 9 | Объем введения | 2-3 стр. машинописного текста |
| 10 | Объем основной части | 15-20 стр. машинописного текста |
| 11 | Объем заключения | 2-2,5 стр. машинописного текста<br>(примерно равен объему введения) |
| 12 | Нумерация страниц | Сквозная, в нижней части листа, посередине. На титульном листе номер страницы не проставляется |
| 13 | Последовательность приведения структурных частей работы | Титульный лист. Задание на курсовую работу. Содержание. Введение. Основная часть. Заключение. Глоссарий. Список использованных источников. Список сокращений. Приложения |
| 14 | Оформление структурных частей работы | Каждая структурная часть начинается с новой страницы. Наименования приводятся с абзаца с прописной (заглавной буквы). Точка в конце наименования не ставится. |
| 15 | Структура основной части | 2-3 главы, соразмерные по объему |
| 16 | Наличие глоссария | Обязательно<br>Не менее 10 |
| 17 | Состав списка использованных источников | Не менее 10 библиографических описаний документальных и литературных источников |
| 18 | Наличие приложений | Обязательно |
| 19 | Оформление содержания (оглавления) | Содержание (оглавление включает в себя заголовки всех разделов, глав, параграфов, глоссария, приложений с указанием страниц начала каждой части |

**ПРИЛОЖЕНИЕ Б**

**Примерный план подготовки курсовой работы**

| № п/п | Наименование этапа | Срок выполнения |
|---|---|---|
| 1 | Сбор необходимой литературы, подготовка библиографического списка. Работа в ТКДБ, IP-системе | 1-2-я недели семестра |
| 2 | Изучение и анализ источников и литературы | 3-5-я недели семестра |
| 3 | Подготовка обзора источников и литературы | 6-8-я недели семестра |
| 4 | Консультации в системе IP- хелпинг | |
| 5 | Формулирование основных теоретических положений и изложение основной части курсовой работы. Консультирование в системе IP- хелпинг | 7-9-я недели семестра |
| 6 | Подготовка введения, заключения | 10-я неделя семестра |
| 7 | Оформление курсовой работы и приложений | 11-12-я недели семестра |
| 8 | Представление курсовой работы для проверки ПО Нормоконтроль. Консультирование в системе IP- хелпинг | 13-я неделя семестра |
| 9 | Внесение исправлений и дополнений по замечаниям | 14-я неделя семестра |
| 10 | Сдача курсовой работы в базовый вуз | 15-я неделя семестра |

**ПРИЛОЖЕНИЕ В**

**Шаблон для формирования, транспортировки и хранения**
**курсовой работы/ курсового проекта**

**Основные данные о работе**

| | |
|---|---|
| Версия шаблона | 2.1 |
| Филиал | |
| Вид работы | Курсовая работа |
| Название дисциплины | |
| Тема | |
| Фамилия студента | |
| Имя студента | |
| Отчество студента | |
| № контракта | |

Здесь и ниже приведены рекомендации по заполнению шаблона «Курсовая работа».

Внимание! Шаблон «Курсовая работа» отформатирован в соответствии требованиями по оформлению курсовой работы. Рекомендуем Вам не менять форматирование шаблона.

После выполнения курсовой работы Вы должны удалить текст рекомендаций, выделенный синим цветом.

Заполните таблицу основных данных о работе.

Обязательные для заполнения поля:

Поле «Версия шаблона» – данное поле должно содержать значение версии заполняемого шаблона. Менять в поле указанную версию шаблона запрещено.

Поле «Филиал» – данное поле должно содержать название филиала.

Поле «Вид работы» – предназначено для ввода вида работы.

Поле «Название дисциплины» – данное поле должно содержать название дисциплины.

Поле «Тема» – данное поле должно содержать тему курсовой работы.

Поле «Фамилия студента» – предназначено для ввода фамилии студента.

Поле «Имя студента» – предназначено для ввода имени студента.

Поле «№ контракта» – предназначено для ввода № контракта.

Необязательные для заполнения поля:

Поле «Отчество студента» – данное поле предназначено для ввода отчества студента.

**Содержание**

Здесь разместите содержание

**Введение**

Здесь разместите текст введения

**Основная часть**

1 глава основной части

Здесь разместите текст первой главы основной части

2 глава основной части

Здесь разместите текст второй главы основной части

3 глава основной части

Здесь разместите текст третьей главы основной части

В заголовках элементов вместо «1 глава основной части», «2 глава основной части», «3 глава основной части» должны быть написаны номера и названия соответствующих глав, заголовок «Основная часть» должен оставаться без изменений.

Если в работе отсутствует элемент «3 глава основной части», заголовок «3 глава основной части» необходимо удалить.

**Заключение**

Здесь разместите текст заключения

**ГЛОССАРИЙ**

| № п/п | Понятие | Определение |
|---|---|---|
| В данной колонке разместите порядковые номера понятий глоссария.<br>Порядковый номер может проставляться как вручную, так и автоматически.<br>Количество строк в таблице должно строго соответствовать количеству внесенных в нее понятий. Пустых строк в таблице быть не должно | В данной колонке разместите колонке понятия глоссария | В данной колонке разместите определения понятий глоссария |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

**Список использованных источников**

| В данной колонке разместите порядковые номера использованных источников. Нумерация использованных источников должна быть сквозная.<br><br>Порядковый номер может проставляться как вручную, так и автоматически.<br><br>Количество строк в таблице должно строго соответствовать количеству внесенных в нее использованных источников. Пустых строк в таблице быть не должно | В данной колонке разместите библиографические описания использованных источников |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

**Список сокращений**

Здесь разместите список сокращений.

Если в работе отсутствует элемент «Список сокращений», заголовок «Список сокращений» необходимо удалить.

**Приложения**

| | |
|---|---|
| Здесь разместите порядковую букву приложения. Нумерация приложений должна быть сквозная, за исключением букв Ё, З, Й, О, Ч, Ь, Ы, Ъ.<br><br>Порядковый номер может проставляться как вручную, так и автоматически.<br><br>Количество строк в таблице должно строго соответствовать количеству приложений. Пустых строк в таблице быть не должно | Здесь разместите файл приложения |

**ПРИЛОЖЕНИЕ Г**

**Образец оформления глоссария**

**ГЛОССАРИЙ**

| № п/п | Новое понятие | Содержание |
|---|---|---|
| 1 | **Облигация** | ценная бумага, подтверждающая обязательство возместить ее владельцу номинальную стоимость с уплатой фиксированного процента |
| 2 | **Патент** | документ, удостоверяющий государственное признание техниче-ского решения изобретением и закрепляющий за лицом, которому он выдан, исключительное право на изобретение |
| 3 | **…..** | ……. |
| 4 | **…..** | ……. |
| 5 | **…..** | ……. |

**ПРИЛОЖЕНИЕ Д**

АТТЕСТАЦИОННЫЙ ЛИСТ (образец)

**КУРСОВОЙ РАБОТЫ СТУДЕНТА**

Студент _____
(Фамилия, имя, отчество)

Учебный план _____

Дисциплина _____

Тема _____

№ контракта _____

**Аттестация выполнена по действующей методике, утвержденной Ученым советом** образовательной организации**, с учетом требований ФГОС ВО по направлению подготовки**

| № п/п | Наименование аттестационного критерия | Описание критерия | Процентное достижение с учетом дисконта |
|---|---|---|---|
| 1 | Самостоятельность | Выявляется степень самостоятельной работы, определяемая относительным количеством цитат из работ других авторов | |
| 2 | Профессионализм | Оценивается уровень сформированности профессиональных компетенций | |
| 3 | Актуальность | Оценивается использование современных достижений в рассматриваемой предметной области, действующего законодательства, актуального фактологического материала | |
| 4 | Общий культурный уровень | Оценивается уровень сформированности общекультурных компетенций. | |
| 5 | Соответствие нормам современного русского языка | Оценивается соблюдение норм литературного языка, количество стилистических ошибок в тексте, уровень грамотности. | |

**Интегральное процентное достижение с учетом весовых значений критериев – …… %**
**Оценка работы в четырехбалльной системе: ……..**
**Дата проведения аттестации: «___» _____ 20__ года**
**Профессорско-преподавательский состав**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**КУРСОВАЯ РАБОТА**

ПОРЯДОК НАПИСАНИЯ, ОФОРМЛЕНИЯ И СДАЧИ

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.Н. Горбатова
Оператор компьютерной верстки А.В. Митряхина

**4184.Курс.01;МПК.01;3**

**МЕТОДИЧЕСКОЕ ПОСОБИЕ**
**ПО КУРСОВОЙ РАБОТЕ**

**ПРОГРАММИРОВАНИЕ**

**МОСКВА 2018**

Разработано И.В. Глазыриной, к.п.н., доц.;
Т.А. Лабзиной, доц.
Под ред. А.П. Пятибратова, д.т.н., проф., засл. деятеля науки и техники РСФСР

Рекомендовано Учебно-методическим
советом в качестве учебного пособия
для обучающихся

**МЕТОДИЧЕСКОЕ ПОСОБИЕ**
**ПО КУРСОВОЙ РАБОТЕ**

**ПРОГРАММИРОВАНИЕ**

Изложены основные требования и рекомендации по выполнению курсовой работы. Приводится перечень тем курсовых работ и список литературы.

**О Г Л А В Л Е Н И Е**

Стр.

## 1 ОБЩИЕ ПОЛОЖЕНИЯ

Курсовая работа предусмотрена тематическим планом изучения дисциплины “Программирование”. Курсовая работа является самостоятельной работой обучающихся, позволяет оценить качество знаний и отражает приобретенные обучающимся практические навыки.

Курсовая работа позволяет расширить объем знаний обучающихся в области программирования и создать реальную основу использования своих знаний для решения на ЭВМ задач по другим дисциплинам и в своей дальнейшей практической деятельности.

Тема назначается руководителем курсовой работы и утверждается на заседании учебно-методической комиссии.

Перед обучающимся ставится задача разработать приложение для Windows с целью решения конкретной задачи. Результатом решения являются:

а) пояснительная записка, составленная с учетом требования стандартов ЕСПД;

б) исполняемый файл программы, прилагаемый к пояснительной записке на диске/дискете.

Для решения поставленной задачи обучающемуся необходимо предварительно ознакомиться с литературой, посвященной теме задания. При этом следует обратить внимание на средства, используемые для решения аналогичных задач или для решения каких-либо ключевых моментов задачи. Этап работы с литературой должен закончиться обзором, в котором собраны полученные сведения из литературы, дан их анализ с точки зрения приложения к поставленной задаче.

Сформулированные в настоящем пособии задания на курсовую работу представляют обучающемуся простор для творчества. В текстах задач умышленно опущены некоторые детали и необходимые требования. После ознакомления с литературой обучающийся должен оценить возможности языка программирования и вычислительной техники, на которой предлагается реализовать решение. Результатом этой работы должна быть точная формулировка задачи со всеми ограничениями и требованиями.

При решении задачи необходимо придерживаться техники пошаговой детализации, использовать стандартные структуры, не забывая при этом о развитии программного окружения программиста, расширяя возможности языка за счет включения новых процедур и функций.

При разработке алгоритма необходимо предусмотреть средства проверки и тестирования программы, удобство работы пользователя, возможные модификации.

При написании программы не следует забывать о хорошем стиле программирования, о таких понятиях, как читабельность, эффективность, надежность. Необходимо искать наиболее простые и естественные приемы и методы решения.

В программе, кроме решения непосредственно задачи, обучающийся должен предусмотреть вывод справки о программе и информации о разработчике с указанием ФИО, группы и даты разработки.

На диске вместе с программой должны быть представлены файлы, подготовленные для проверки ее работоспособности.

## 2 ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

Выполнение курсовой работы состоит из трех этапов.

1. Подготовительный этап (разработка эскизного и технического проектов).
2. Практическая работа за компьютером (разработка рабочего проекта).
3. Оформление пояснительной записки.

## 3 СОДЕРЖАНИЕ РАЗДЕЛОВ КУРСОВОЙ РАБОТЫ

Все этапы разработки программы отражаются в пояснительной записке.

Пояснительная записка состоит из следующих разделов:

Введение

1 Разработка эскизного и технического проектов программы (ГОСТ 19.404–79)

1.1 Задание

1.2 Назначение и область применения

1.3 Технические характеристики

1.4 Источники, использованные при разработке

2 Разработка рабочего проекта

2.1 Разработка программы

2.2 Спецификация программы

2.3 Текст программы

2.4 Описание программы

2.5 Тестирование программы

3 Внедрение

Заключение

Глоссарий[1]

Список используемой литературы.

При написании пояснительной записки необходимо придерживаться требований единой системы программной документации (ЕСПД) и методических указаний по выполнению письменной курсовой работы, принятых в образовательной организации.

[1] Глоссарий – толковый (объясняющий) словарь понятий и терминов.

### 3.1 Оглавление

Оглавление составляется в соответствии с содержанием пояснительной записки и должно отражать все разделы курсовой работы. После написания пояснительной записки в оглавлении проставляются страницы.

### 3.2 Введение

Во введении кратко характеризуется проблема, решению которой посвящена курсовая работа, определяются цель и задачи, которые надо решить для раскрытия темы, описываются средства, посредством которых реализуется разрабатываемая программа.

### 3.3 Разработка эскизного и технического проектов программы

Стандарт ГОСТ 19.404-79 устанавливает требования к содержанию и оформлению программного документа “Пояснительная записка”, входящего в состав документов на стадиях разработки эскизного и технического проектов программы.

#### 3.3.1 Задание

В разделе “Задание” указывается тема курсовой работы и приводится условие решаемой задачи.

Пример

Тема: Разработка приложения для Windows, представляющего собой компьютерную игру “Лабиринт”.

Условие задачи:

Игра “Лабиринт” состоит в том, что играющий перемещается в двухмерном пространстве по помещениям здания, план которого играющему неизвестен. Начиная с произвольного помещения, путешественник должен найти выход из здания. Каждое помещение может иметь четыре двери: север, восток, юг, запад. План здания необходимо считать из текстового файла в связанный список. Порядок следования помещений в списке должен быть произвольным. Находясь в N-ом помещении, игрок может получить подсказку о правильном направлении движения, если верно выполнит тестовое задание по теме “Программирование на языке высокого уровня”.

#### 3.3.2 Назначение и область применения

В разделе “Назначение и область применения” указывают назначение программы и краткую характеристику области применения программы.

В приведенном примере, например, необходимо разработать развлекательную программу, представляющую собой игру. Область применения: досуг пользователя. Поскольку ставится задача разработать приложение для Windows, то использоваться программа может только под управле-нием ОС Windows.

#### 3.3.3 Технические характеристики

Раздел “Технические характеристики” должен содержать следующие подразделы:

1 Постановка задачи

2 Описание алгоритма

3 Организация входных и выходных данных

4 Выбор состава технических и программных средств.

##### 3.3.3.1 Постановка задачи

Решение задачи начинается с ее постановки. Постановка задачи - точная формулировка решения задачи на компьютере с описанием входной и выходной информации. Входная информация по задаче – это данные,

поступающие на вход задачи и используемые для ее решения. Выходная информация может быть представлена в виде документов, кадров на экране монитора, информации в базе данных, выходного сигнала устройству управления.

В данном разделе дается точное описание исходных данных, условий задачи и целей ее решения. На этом этапе условия задачи, записанные в форме различных словесных описаний, необходимо выразить на формальном языке математики. Обычно математическая модель – это набор уравнений, неравенств и ограничений, приближенно описывающих задачу. При построении математической модели отбрасываются некоторые свойства реальной задачи, мало влияющие на решение.

В этом разделе могут быть описаны основные приемы программирования и типы данных, используемые при решении аналогичных задач. Например, если в задаче используются динамические структуры, то перечисляются виды динамических структур данных и основные процедуры по работе с динамическими структурами. Если задача заключается в формировании базы данных и дальнейшей работе с базой, то приводится описание используемых типов данных (характеристика данных записного типа) и приемы работы с файлами.

Далее описываются возможные пути решения задачи с указанием их достоинств и недостатков. Выбирается и обосновывается метод решения задачи. Описываются ограничения, накладываемые на исходные данные, необходимая разрядность и точность представления исходных данных и результатов решения. Указываются возможные пределы изменения входных параметров задачи.

*Пример*

Разработать программу, моделирующую игру «Лабиринт», с использованием динамической структуры «связанный список». Играющий должен найти выход из двумерного лабиринта. Лабиринт состоит из смежных комнат. Каждая комната может иметь четыре двери, направленные на север, юг, запад и восток. Одна из комнат содержит выход из лабиринта. Играющий, попав в очередную комнату, может попросить подсказку о направлении движения в сторону выхода из лабиринта. Подсказка представляет собой вопрос теста и четыре варианта ответа по дисциплине «Программирование на языке высокого уровня». Правильный ответ соответствует двери, ведущей к выходу. Представляемые играющему вопросы и варианты ответа должны считываться из файла, хранящегося на диске.

После описания общих положений тематики работы, необходимо указать конкретные методы решения поставленной задачи. Математическая формулировка в данном случае заменяется словесным описанием.

**Предположим, что помещения здания соединяются между собой так, как показано на рисунке 1.**

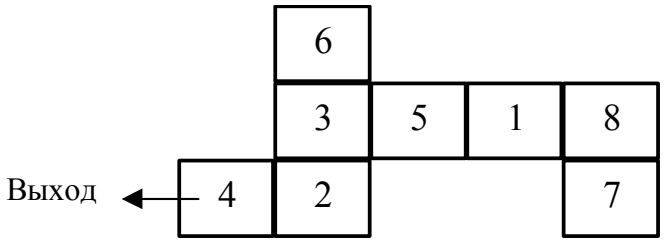



Рисунок 1. Пример плана помещений

Информация о плане здания должна храниться в файле в последовательности: номер комнаты; номера комнат, смежных с данной. Смежные комнаты перечислены в порядке: север, восток, юг, запад (рисунок 2). Порядок следования помещений в этом списке может быть любым, как и последовательность номеров помещений на плане.

| Номер комнаты | Дверь в комнату норд | ост | зюйд | вест |
|---|---|---|---|---|
| 7 | 8 | 0 | 0 | 0 |
| 1 | 0 | 8 | 0 | 5 |
| 8 | 0 | 0 | 7 | 1 |
| 5 | 0 | 1 | 0 | 3 |
| 6 | 0 | 0 | 3 | 0 |
| 3 | 6 | 5 | 2 | 0 |
| 2 | 3 | 0 | 0 | 4 |
| 4 | 0 | 2 | 0 | 999 |
| 999 | 0 | 0 | 0 | 0 |

(0 = нет двери. 999 = выход)

**Рисунок 2. Информация о плане здания, представленном на рисунке 3**

Для последующей обработки файл с данными считывается и помещается в список, исходя из номера помещения (рисунок 3). Spi – указатель на начало списка.

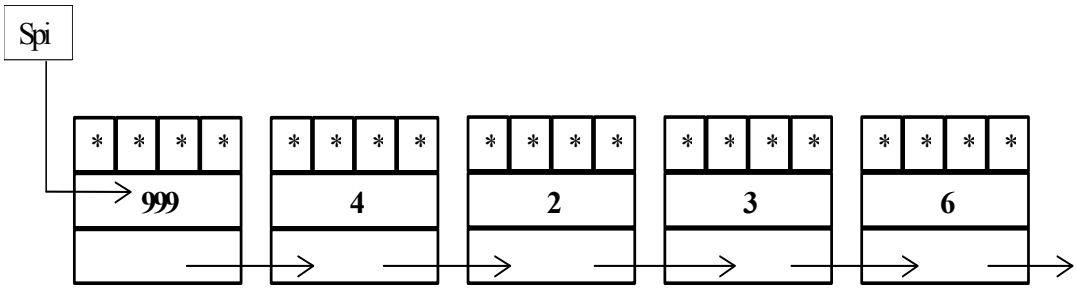



**Рисунок 3. Начало списка помещений после первого прочтения, представленного на рисунке 2 файла**

Затем файл считывается еще раз и с учетом наличия дверей генерируется связь помещений (рисунок 4).

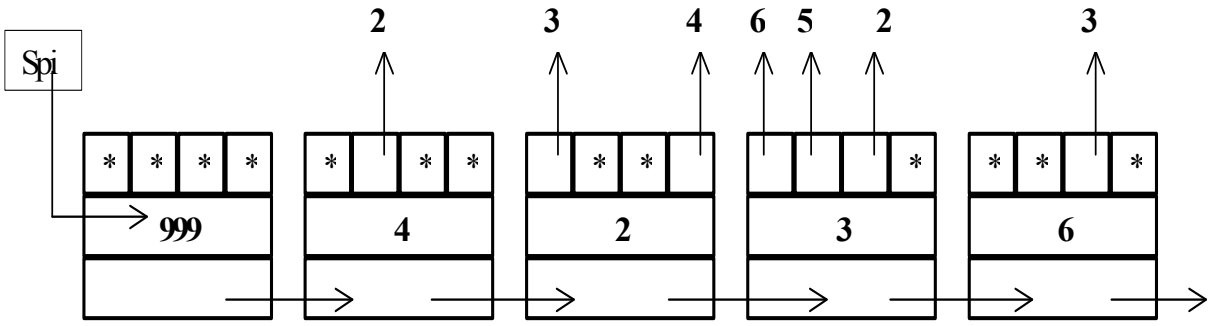



**Рисунок 4. Начало списка помещений после второго прочтения, представленного на рисунке 4 файла**

Звездочкой помечены ссылки, равные Nil.

Перемещение игрока в лабиринте осуществляется путем выбора им одного из вариантов направления движения. Игрок вводит номер комнаты, с которой начинается путешествие. Если такого помещения в лабиринте нет, то об этом выдается сообщение. Если такая комната существует, то играющему предлагается

выбрать направление движения, которое можно определить, воспользовавшись подсказкой. Помощь заключается в том, что на экран высвечивается вопрос по теме курса “Программирование на языке высокого уровня” и 4 варианта ответа. Правильный ответ указывает направление дальнейшего движения. Неправильный ответ дает неверное направление или вызывает сообщение о невозможности передвижения. Вывод подсказки на экран и выбор правильного ответа осуществляются с помощью специальной процедуры, которая обращается к файлу на диске.

Признаком окончания игры является ссылка на помещение с номером 999.

В программе необходимо предусмотреть возможность создания и корректирования файлов плана помещения и подсказки.

#### 3.3.3.2 Описание алгоритма

В разделе дается обобщенное словесное описание алгоритма решения поставленной задачи, излагаются основные требования к алгоритму и пути их реализации. Приводится схема алгоритма, состоящая из укрупненных модулей. Дается пояснение назначения и состава каждого модуля. Обобщенный алгоритм обычно использует обозначения и термины исходной задачи.

На следующем этапе каждый модуль детализируется. Выделяются укрупненные команды, реализуемые по вспомогательным алгоритмам. Тот же подход применяется при разработке вспомогательных алгоритмов.

Пример

В программе решаются три независимые друг от друга подзадачи.

1. Перемещение по лабиринту.
2. Запись на диск плана помещений здания.
3. Запись на диск файла подсказки.

Задача “Перемещение по лабиринту” решается в два этапа:

1) формирование списка помещений;

2) выбор направления движения и переход из одного помещения в другое.

Обобщенная схема алгоритма приведена на рисунке 5.

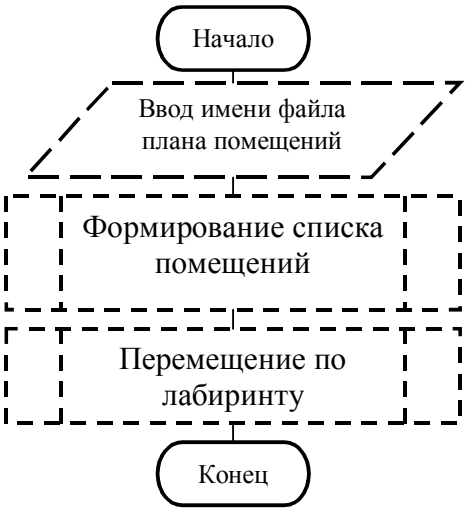



Рисунок 5. Обобщенная схема алгоритма

**При формировании списка плана помещений сначала производятся считывание номеров комнат из текстового файла и построение связанного списка, затем построение многосвязанного списка, содержащего в себе план помещений здания (рисунок 6).**

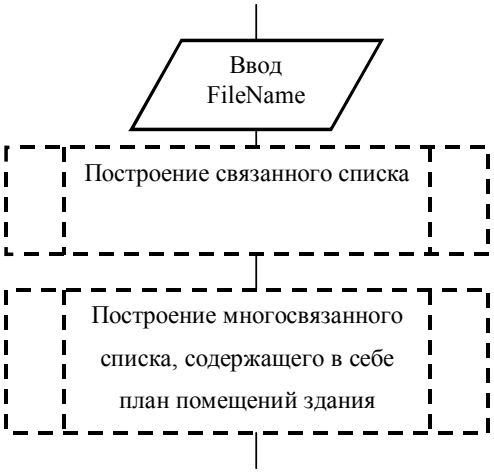



**Рисунок 6. Формирование списка плана помещений**

Перемещение по лабиринту начинается с выбора номера помещения для начала путешествия. Если такого помещения нет, то игра заканчивается. Если такое помещение в лабиринте есть, то игроку предоставляется возможность выбрать направление движения или воспользоваться файлом подсказки. Укрупненная схема алгоритма приведена на рисунке 7.

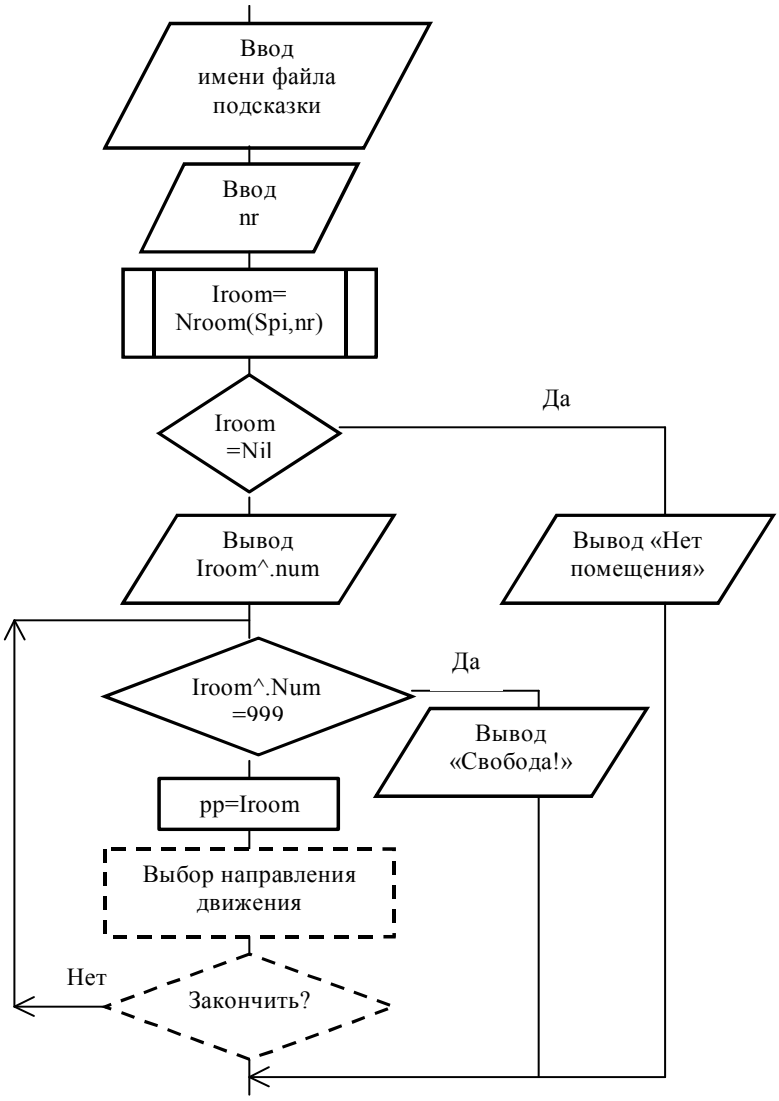



**Рисунок 7. Схема алгоритма этапа “Перемещение по лабиринту”**

**Блок “Выбор направления движения” может быть детализирован на этапе разработки программы.** Аналогично детализируются все составляющие алгоритма.

**На этапе разработки технического проекта детализация на уровне операторов программы необязательна.**

### 3.3.3.3 Организация входных и выходных данных

Данный раздел содержит описание и обоснование выбора метода организации входных и выходных данных.

Файл, содержащий план лабиринта, может быть организован непосредственно в приложении. Для этого разрабатывается отдельный фрагмент программы. Структура файла приведена на рисунке 2. В текстовом файле в одной строке записаны номер комнаты и номера комнат смежных помещений. Смежные помещения располагаются в порядке: норд, ост, зюйд, вест. Если смежное помещение отсутствует, то вводится значение 0. Номера комнат в файле могут располагаться в произвольном порядке. Важным является то, что информация о комнате должна быть расположена в отдельной строке.

Разрабатываемое приложение предусматривает использование файлов прямого доступа. В файлах прямого доступа хранится информация подсказки. Подсказкой является правильный ответ тестового задания, который соответствует двери, ведущей к выходу из лабиринта. Файл–помощь может быть сформирован непосредственно в приложении. Для этого разрабатывается отдельный фрагмент программы. Количество компонентов файла равно количеству помещений в здании-лабиринте. Номер компоненты файла соответствует номеру помещения. Правильный ответ расположен в положении, определяющем верное направление. Компонент типизированного файла имеет тип “запись”, где отводится поле для хранения вопроса по теме курса и поле – массив предлагаемых ответов.

Структура файла подсказки:

Запись:

Поле вопроса

Поле 4–х ответов

Файл подсказки для структуры помещений (см. рисунок 1) может содержать, например, следую-щие тестовые задания:

Что является стандартной файловой переменной?

Con

Prn

F

Input

Правильно описана файловая переменная текстового типа

F4: text80;

F1: File of String;

F: File of String[80];

F2: Text;

Какая процедура предназначена для связи файловой переменной с физическим файлом на диске?

Reset

Rewrite

Assign

Read

Какая процедура предназначена для чтения данных из типизированного файла?

Readln

ReadBlock

Input

Read

Если Var a, b: ^Real; то для a и b справедливо
New(a^)
a:= a/b
a^:= a^ mod b^
a^:= sin(b^)
В результате выполнения New(p) p приобретает значение, соответствующее
значению ноль
типу переменной p
адресу, начиная с которого можно разместить данные
значению Nil
и т.д.

#### 3.3.3.4 Выбор состава технических и программных средств

На основании разработанного алгоритма делается вывод о необходимости использования того или иного языка программирования. Перечисляются достоинства выбранной среды программирования. Определяются технические средства, необходимые для оптимальной работы будущей программы.

#### 3.3.3.5 Источники, использованные при разработке

Данный раздел должен присутствовать в пояснительной записке, если при разработке прило-жения использовались другие готовые программы.

### 3.4 Разработка рабочего проекта

Этап разработки рабочего проекта включает в себя разработку программы и программной документации, а также испытание программы.

#### 3.4.1 Разработка программы

Современные программы разрабатываются для функционирования в среде Windows. Приложение для Windows обучающиеся разрабатывают в среде визуального программирования. Визуальное программирование строится на тесном взаимодействии двух процессов:

- процесса конструирования Windows-окна;
- процесса написания кода, придающего элементам этого окна и программе в целом необхо-димую функциональность.

Проект Windows-окна должен быть представлен в виде графической схемы, на которой распо-ложены все визуальные и невизуальные компоненты, разрабатываемого интерфейса. Компоненты на схеме должны быть пронумерованы. После схемы приводится расшифровка изображенных на схеме компонентов: название и имя компонента; назначение в программе; события, на которые данный компонент откликается. Для каждого компонента должны быть указаны свойства, изме-ненные при проектировании окна.

Пример

Для разработки приложения игры “Лабиринт” используется среда визуального программирования. Проект программы содержит три окна:

1. Form1 – перемещение по лабиринту.

2. Form2 – запись на диск плана помещений здания.

3. Form3 – запись на диск файла подсказки.

Окно Form1 Перемещение по лабиринту (рисунок 8).


Рисунок 8. Главное окно программы – Form1

Компонент Form1

Свойства:

Caption – Игра “Лабиринт”;

Border – bsSingle;

BiSystemMenu – false;

BiMinimaze – false;

BiHelp – False;

События: нет.

1 – компонент TMainMenu

Свойства:

Items

Игра (N1) Настройка (N2) Выход (N3)

Ввод плана помещения (N4)

Ввод файла помощи (N5)

**События:**

**Для N1 – событие BitBtn1Click (приводится ниже).**

N3Click – закончить работу с приложением.

N4Click – перейти к работе со второй формой.

**N5Click – перейти к работе с третьей формой.**

2 – компонент TLabel1

**Свойства:**

Font – сиреневый, размер 12.

3 – компонент TEdit1

Свойства:

Text – очистить.

События:

**Edit1KeyPress – защита от ввода недопустимых символов.**

4 – компонент TBitBtn1

Свойства:

Caption – Game;

Glyph – Picture.Bmp.

События:

BitBtn1Click – ввод плана помещений из файла и организация структуры “Связанный список”. Выводит сообщение в метку Label1 и делает видимыми: Buttun1, Label1, Edit1.

5 – компонент TBitBtn2

**Свойства:**

Kind – bkClose

6 – компонент TPanel1

Свойства:

Align – alClient;

Caption – очистить;

Visible – False.

7 – компонент TPanel2

Свойства:

Align – alClient;

Caption – очистить;

Visible – False.

Компонент Image1 (на схеме не пронумерован).

Свойства:

Picture – Arches.Bmp;

Stretch – True;

Компонент Button1 (закрыт компонентом BitBtn1);

Свойства:

Caption – продолжить.

События:

Button1Click – ввод номера помещения, ввод имени файла помощи, показывает Panel2, прячет Edit1, Label1, Button1.

Контейнер Panel1 (рисунок 9).

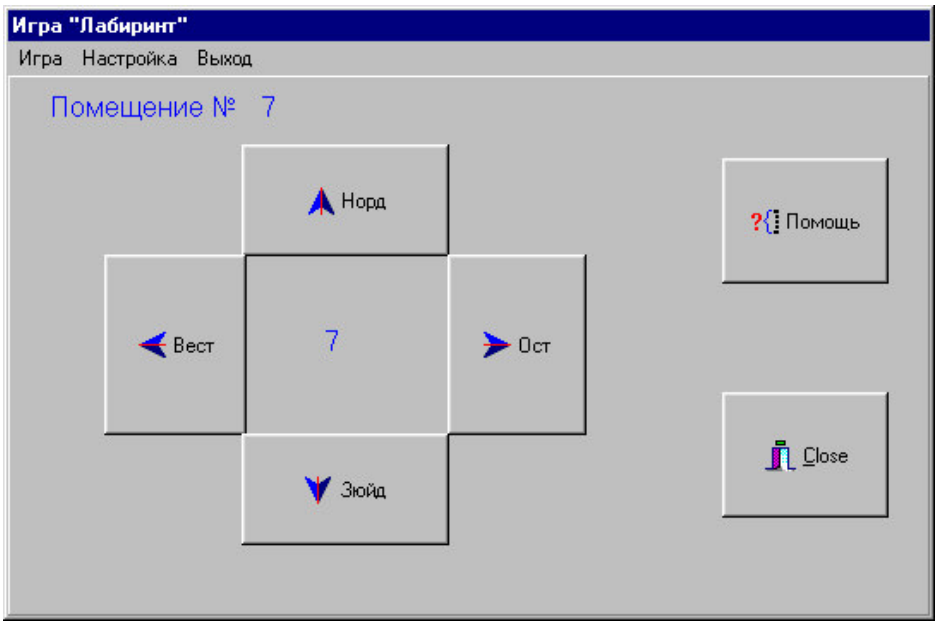



Рисунок 9. Компонент Panel1

Перечисляются компоненты, расположенные в контейнере Panel1, их свойства и события.

Далее приводятся графические изображения других панелей и окон, перечисляются компоненты и их свойства.

Написание программы

Программную реализацию разработанных алгоритмов содержат обработчики событий. На этапе разработки рабочего проекта необходимая степень детализации алгоритмов обычно выбирается такой, чтобы предписания разработанных алгоритмов могли записываться на языке программирования, выбранном для составления текста программы. При детализации алгоритма необходимо перейти к обозначениям, принятым для разработки программ на алгоритмическом языке. При этом имена следует выбирать таким образом, чтобы они отражали сущность используемых параметров.

Кодирование должно быть простым. Изощренное программирование может обойтись слишком дорого при отладке и модификации программы. Необычное кодирование (например, использование скрытых возможностей машины) часто препятствует отладке программы и затрудняет ее использование другими программистами.

Программа должна быть по возможности универсальной. Универсальные программы обеспечивают независимость программы от конкретного набора данных. Например, универсальная программа использует в качестве параметров переменные, а не константы, обрабатывает вырожденные случаи и т.д.

Универсальность программы экономит время по дальнейшей работе над ней и обеспечивает широкие возможности по использованию. Разрабатывая такие программы, можно предвидеть будущие изменения в спецификациях этой программы.

Входные форматы должны быть разработаны с учетом максимального удобства для пользователя и минимальной возможности ошибок. Порядок переменных и форматы данных, привычные для пользователя, помогут избежать ошибок и облегчат использование программ.

При написании программы следует применять операторы, позволяющие использовать основные алгоритмические структуры. Оператор Goto желательно не использовать.

При написании программ не следует забывать о хорошем стиле программирования. После заголовка процедуры или функции записывается комментарий, содержащий поясняющий текст, а именно: назначение подпрограммы; перечень и назначение формальных параметров, их тип. Комментариями должны быть снабжены и основные смысловые блоки программы или подпрограммы.

Для облегчения чтения текста программы отдельные операторы программы записываются с отступом.

Пример

Обработчик BitBtn1Click.

Для построения детальной схемы алгоритма необходимо определить структуру элемента списка и ввести обозначения.

Предлагается следующая структура элемента списка помещений:

Room = Record Num: Integer;

Door: Array [Trend] Of Uk;

Next: Uk;

End;

Поле Num хранит номер помещения; массив Door предназначен для хранения ссылок на смежные комнаты; поле Next содержит ссылку на следующий элемент списка. Trend – это перечисляемый тип данных, имеющий следующие значения: nord, ost, zued, west. Эти значения соответствуют направлениям дверей, которые могут быть расположены в каждом помещении. Указатель на начало списка помещений обозначим идентификатором Spi. Spi – глобальная переменная, которая описана в головной части программы.

Схема алгоритма построения связанного списка приведена на рисунке 10.

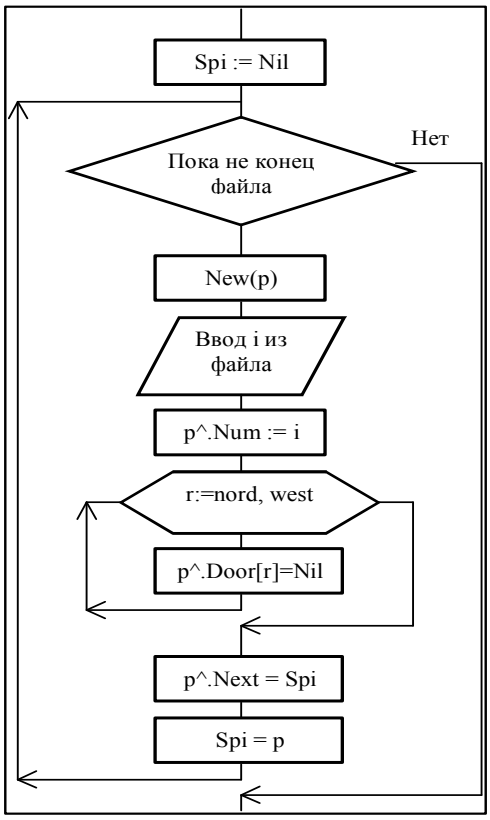



**Рисунок 10. Построение связанного списка**

**Схема алгоритма построения многосвязанного списка приведена на рисунке 11.**

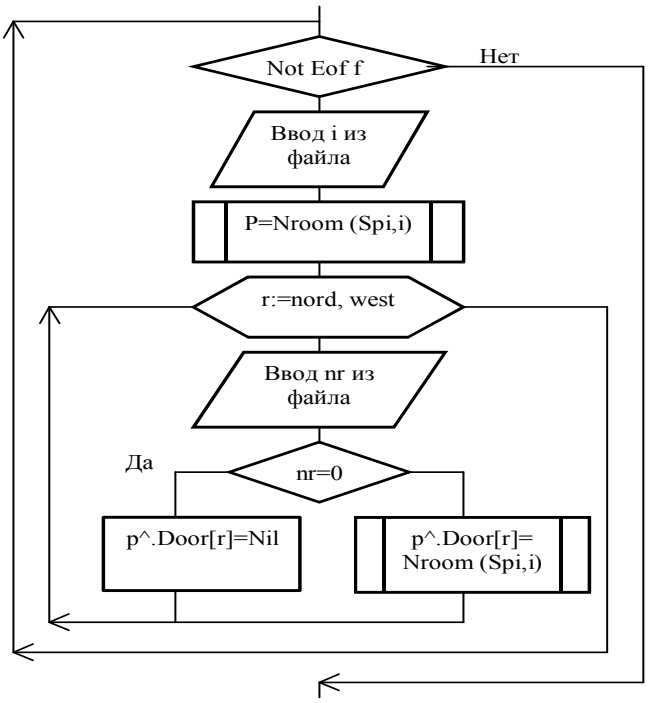



**Рисунок 11. Схема алгоритма построения многосвязанного списка, содержащего в себе план помещений здания**

**Программа**

```
procedure TForm1. BitBtn1Click (Sender: TObject);
//Построение связанного списка помещений
Var FileName: TNameFile; f: TextFile;
p: Uk; i, nr: Integer;
begin
FileName:= InputBox('План помещений', 'Введите имя файла ','');
AssignFile(f, FileName);
Reset(f);
//Считываем только номера помещений и составляем связанный список
Spi:= nil;
While not Eof(f) Do
Begin
New(p);
Readln(f, i);
p^.Num:= i;
For r:= nord To west Do p^.Door [r]:= nil;
p^.next:= Spi;
Spi:= p;
End;
//Еще раз считываем из файла, но уже с дверьми
Reset (f);
While not Eof(f) Do
Begin
Read(f, i);
p:= Nroom(Spi, i);
For r:= nord To west Do
Begin Read(f, nr);
```

```
If nr = 0 Then p^.Door[r]:= nil
Else p^.Door[r]:= Nroom(Spi, nr);
End;
End;
//Подготовка формы для ввода номера начала помещения
label1.Caption:= ‘Введите № комнаты, с которой хотите начать движение’;
Edit1.Visible:= True;
label1.Visible:= True;
Button1.Visible:= True;
end;
```

**Функция Nroom – поиск ссылки на помещение с заданным номером. В начинающемся в Spi списке помещений по номеру помещения N отыскивается нужное. Значение функции указывает на это помещение. Схема алгоритма приведена на рисунке 12.**

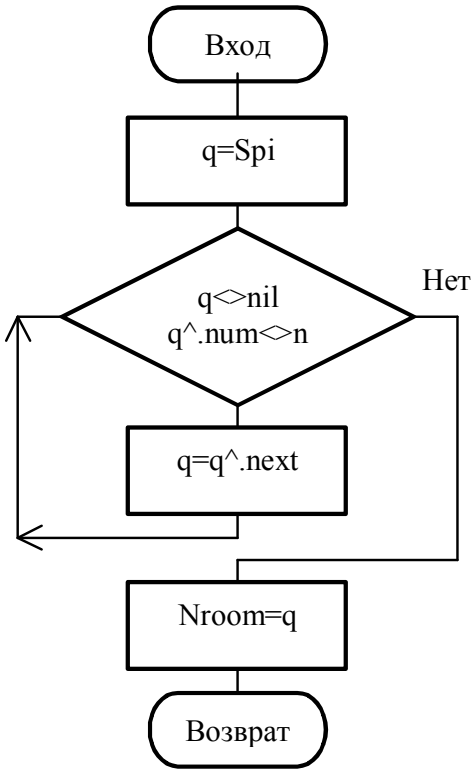



**Рисунок 12. Схема алгоритма функции Nroom**

```
Программа
Function Nroom(Spi: Uk; n: Integer): Uk;
//В списке помещений находим указатель на n-ое
Var q: Uk;
Begin
q:= Spi;
While (q<>nil) and (q^.num<>n) Do q:= q^.next;
Nroom:= q;
End;
```

Аналогично уточняются все алгоритмы и разрабатываются обработчики событий.

**Примечание - не следует представлять в виде схем алгоритмов линейные вычислительные процессы. В детализированном виде представляются алгоритмы, имеющие сложную структуру. Сложной структурой будем считать алгоритмы, содержащие более одного разветвления или более одного цикла.**

### 3.4.2 Спецификация программы

В разделе “Спецификация” приводится точное название программы и ее состав. Форма спецификации приведена в приложении А. Графы спецификации заполняют следующим образом:

- в графе “Обозначение” указывают обозначение основных программных компонентов;

- в графе “Наименование” - полное наименование соответствующего компонента;

- в графе “Примечание” – дополнительные сведения, относящиеся к записанным в спецификации программам.

Пример

Исполняемый файл программы Игра “Лабиринт” имеет название Labirint.exe и расположен на компакт-диске (дискете) в каталоге Kursov\Labirint. Состав проекта.

| Наименование | Обозначение | Примечание |
|---|---|---|
| Ad | Файл плана помещений | Создается в приложении |
| Help 1 | Файл помощи | Создается в приложении |
| Labirint.Dof | Файл параметров проекта | Содержит текущие установки проекта: настройки компилятора и компоновщика, имена служебных каталогов, условные директивы |
| Labirint.Dpr | Файл проекта | Связывает все файлы, из которых состоит приложение |
| Labirint.Dsk | Файл, содержащий Desktop – настройки проекта | Содержит информацию о том, какие окна открыты и в каких позициях они расположены |
| Labirint.Res | Файл ресурсов | Содержит пиктограммы, графические изображения |
| Unit1.Pas | Файл программного модуля для формы № 1 | Определяет функциональность формы № 1 |
| Unit2.Pas | Файл программного модуля для формы № 2 | Определяет функциональность формы № 2 |
| Unit3.Pas | Файл программного модуля для формы № 3 | Определяет функциональность формы № 3 |
| Unit1.Dfm | Файл формы № 1 | Содержит список свойств всех компонентов, включенных в форму № 1 |
| Unit2.Dfm | Файл формы № 2 | Содержит список свойств всех компонентов, включенных в форму № 2 |
| Unit3.Dfm | Файл формы № 3 | Содержит список свойств всех компонентов, включенных в форму № 3 |
| Unit1.Dcu | Объектный файл для Unit1.Pas | Откомпилированная версия Unit1.Pas |
| Unit2.Dcu | Объектный файл для Unit2.Pas | Откомпилированная версия Unit2.Pas |
| Unit3.Dcu | Объектный файл для Unit3.Pas | Откомпилированная версия Unit3.Pas |

### 3.4.3 Текст программы

Приводится полный листинг программы.
Пример листинга:

```
unit Unit1;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, Buttons, Menus, ExtCtrls;
type
```

```
TForm1 = class(TForm)
Image1: TImage;
MainMenu1: TMainMenu;
N1: TMenuItem;
N2: TMenuItem;
N3: TMenuItem;
BitBtn1: TBitBtn;
BitBtn2: TBitBtn;
Panel1: TPanel;
Label1: TLabel;
Edit1: TEdit;
Label2: TLabel;
Label3: TLabel;
BitBtn3: TBitBtn;
Label4: TLabel;
BitBtn4: TBitBtn;
BitBtn5: TBitBtn;
BitBtn6: TBitBtn;
BitBtn7: TBitBtn;
BitBtn8: TBitBtn;
Panel2: TPanel;
Memo1: TMemo;
ListBox1: TListBox;
Button1: TButton;
N4: TMenuItem;
N5: TMenuItem;
Label5: TLabel;
procedure BitBtn1Click(Sender: TObject);
procedure Edit1KeyPress(Sender: TObject; var Key: Char);
procedure BitBtn3Click(Sender: TObject);
procedure BitBtn7Click(Sender: TObject);
procedure ListBox1Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure N4Click(Sender: TObject);
procedure N3Click(Sender: TObject);
procedure N5Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var Form1: TForm1;
//Перечисляемый тип для определения направления движения
Type Trend = (nord, ost, zued, west);
//Структура элемента списка
Uk = ^Room;
Room = Record Num: Integer;
Door: Array [Trend] Of Uk;
Next: Uk;
End;
//Структура компонента файла помощи
THelp = Record v: String[100];
otv: Array [1..4] Of String[100];
End;
TNameFile = String[20];
Var Spi, pp, Iroom: Uk;
r: Trend; ff: File Of THelp;
//Spi – указатель на начало списка; pp, Iroom – рабочие указатели;
//r – для определения направления движения;
//ff – файловая переменная для обращения к файлу помощи.
implementation
Uses Unit2, Unit3;
{$R *.DFM}
```

```
Function Nroom(Spi: Uk; n: Integer): Uk;
//В списке помещений находим указатель на n-ое
Var q: Uk;
Begin
q:= Spi;
While (q<>nil) and (q^.num<>n) Do q:= q^.next;
Nroom:= q;
End;

procedure TForm1.BitBtn1Click(Sender: TObject);
//Построение связанного списка помещений
Var FileName: TNameFile; f: TextFile;
p: Uk; i, nr: Integer;
begin
FileName:= InputBox('План помещений','Введите имя файла ','');
AssignFile(f, FileName);
Reset(f);
//Считываем только номера помещений и составляем связанный список
Spi:= nil;
While not Eof(f) Do
Begin
New(p);
Readln(f, i);
p^.Num:= i;
For r:= nord To west Do p^.Door [r]:= nil;
p^.next:= Spi;
Spi:= p;
End;
//Еще раз считываем из файла, но уже с дверьми
Reset (f);
While not Eof(f) Do
Begin
Read(f, i);
p:= Nroom(Spi, i);
For r:= nord To west Do
Begin Read(f, nr);
If nr = 0 Then p^.Door[r]:= nil
Else p^.Door[r]:= Nroom(Spi, nr);
End;
End;
//Подготовка формы для ввода номера начала помещения
label1.Caption:= 'Введите № комнаты, с которой хотите начать движение';
Edit1.Visible:= True;
label1.Visible:= True;
Button1.Visible:= True;
end;
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
//Защита от недопустимых символов (вводятся только цифры)
begin
If not (key in ['1'..'9']) Then Key:= #27;
end;

procedure TForm1.BitBtn3Click(Sender: TObject);
//Перемещение. BitBtn3 – север; BitBtn4 – запад;
//BitBtn5 – восток; BitBtn6 – юг.
Var aa: String[8]; c: Integer; pp: Uk;
//aa – имя кнопки; c – номер кнопки;
//pp – указатель текущего помещения.
begin
pp:= Iroom;
aa:= (sender as TBitBtn).Name;
c:= StrToInt(Copy(aa,7,1));
Case c of
```

```
3: r:= nord;
5: r:= ost;
6: r:= zued;
4: r:= west;
End;
Iroom:= Iroom^.Door[r];
If Iroom = nil Then
Begin ShowMessage('Это невозможно!');Iroom:= pp; End
Else Label3.Caption:= IntToStr(Iroom^.Num);
Label4.Caption:= IntToStr(Iroom^.Num);
If Iroom^.Num = 999 Then
Begin ShowMessage('Свобода!!!');
Panel1.hide; label1.Hide; Button1.Hide;
Exit End;
end;
procedure TForm1.BitBtn7Click(Sender: TObject);
//Вывод содержимого справки в компоненты панели Panel2
Var i: Integer; a: THelp;
Begin
Panel1.Visible:= False;
Panel2.Visible:= True;
Label5.Caption:= IntToStr(Iroom^.Num);
//Устанавливаем указатель текущего компонента файла
Reset(ff); Seek(ff, Iroom^.Num - 1);
Read(ff, a); Memo1.Lines.Add(a.v);
For i:= 1 To 4 Do ListBox1.items[i]:= a.otv[i];
end;
procedure TForm1.ListBox1Click(Sender: TObject);
//Определение направления путем выбора верного ответа в списке
Var r1: Trend;
begin
pp:= Iroom;
Panel2.Visible:= False;
Panel1.Visible:= True;
for r1:= nord To west Do
If ListBox1.ItemIndex = ord(r1) + 1 Then r:= r1;
Iroom:= Iroom^.Door[r];
If Iroom = nil Then
Begin ShowMessage('Это невозможно!');Iroom:= pp; End
Else Label3.Caption:= IntToStr(Iroom^.Num);
Label4.Caption:= IntToStr(Iroom^.Num);
If Iroom^.Num = 999 Then
Begin Writeln('Свобода!!!');
Panel1.hide; label1.Hide; Button1.Hide;
Exit End;
end;
procedure TForm1.Button1Click(Sender: TObject);
//Ввод номера начала помещения
Var nr: Integer; FileName: TNameFile;
//nr – номер помещения для начала движения
//File Name – имя файла помощи
begin
If Edit1.Text = '' Then Exit;
nr:= StrToInt(Edit1.Text);
Iroom:= Nroom(Spi, nr);
If Iroom = nil Then
Begin
ShowMessage('Такого помещения нет!');
Edit1.Text:= ''; Label1.Caption:= '';
Exit;
End;
panel1.Visible:= True;
Label3.Caption:= IntToStr(Iroom^.Num);
```

```
Label4.Caption:= IntToStr(Iroom^.Num);
FileName:= InputBox('Имя файла помощи', 'Введите имя файла помощи','');
AssignFile(ff, FileName);
Edit1.Hide; Label1.Hide; Button1.Hide;
end;
procedure TForm1.N4Click(Sender: TObject);
//Переход к форме №2, осуществляющей запись плана помещений на диск.
Begin
Form1.Hide; Form2.Show;
End;
procedure TForm1.N3Click(Sender: TObject);
//Закрыть приложение
begin
Form1.Close; Form2.Close; Form2.Close;
Application.Terminate;
end;
procedure TForm1.N5Click(Sender: TObject);
//Переход к форме №3, осуществляющей запись на диск файла помощи.
begin
Form1.Hide; Form3.Show;
end;
end.
```

**Далее приводятся листинги остальных модулей.**

### *3.4.4 Описание программы*

**Раздел "Описание программы" согласно ГОСТ 19.402-78* должен содержать следующие подразделы:**

- общие сведения;
- функциональное назначение;
- описание логической структуры;
- используемые технические средства;
- вызов и загрузка;
- входные данные;
- выходные данные.

Отдельные разделы можно объединять. Некоторые пункты этого раздела повторяют разделы технического проекта. Такие повторения предусмотрены ГОСТом, так как на этапе рабочего проекта возникают некоторые дополнения или изменения в составе технических средств или программе. Здесь приводятся более конкретные и точные данные.

В подразделе "Общие сведения" должны быть указаны: обозначение и наименование программы; программное обеспечение, необходимое для функционирования программы; языки программирования, на которых написана программа.

В подразделе "Функциональное назначение" должны быть указаны классы решаемых задач и (или) назначение программы и сведения о функциональных ограничениях на применение.

В подразделе "Описание логической структуры" должны быть указаны используемые методы; структура программы с описанием функций составных частей и связи между ними; связи программы с другими программами. Описание логической структуры программы выполняют с учетом текста программы на исходном языке.

В подразделе "Используемые технические средства" должны быть указаны типы ЭВМ и устройств, которые используются при работе программы.

В подразделе "Вызов и загрузка" должны быть указаны способ вызова программы с соответствующего носителя данных, входные точки в программу.

В подразделе “Входные данные” должны быть указаны: характер, организация и предварительная подготовка входных данных, формат, описание и способ кодировки входных данных.

В подразделе “Выходные данные” должны быть указаны: характер, организация и предварительная подготовка выходных данных, формат, описание и способ кодировки выходных данных.

### *3.4.5 Тестирование программы*

Перечисляются требования, подлежащие проверке при испытании программы, а также поря-док и методика их контроля. Приводятся исходные данные для решения контрольного примера и ожидаемые результаты.

Прилагаются распечатка решения контрольного примера и снимки экрана с результатами тес-тирования.

### *3.5 Внедрение*

В разделе описываются (руководство оператора ГОСТ 19.505–79):

- условия выполнения программы;
- выполнение программы;
- сообщения оператору.

В разделе “Условия выполнения программы” должны быть указаны условия, необходимые для выполнения программы (минимальный и/или максимальный состав аппаратурных и програм-мных средств и т.п.).

В разделе “Выполнение программы” должна быть указана последовательность действий опе-ратора, обеспечивающих загрузку, запуск, выполнение и завершение программы. В разделе приво-дятся сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения. Перечисляются порядок и последовательность ввода исходных данных и получения результатов расчета.

В разделе “Сообщения оператору” должны быть приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора.

Содержание разделов допускается иллюстрировать поясняющими примерами, таблицами, схемами.

### 3.6 Список использованных источников

В разделе перечисляется литература, использованная при выполнении курсовой работы. В тексте пояснительной записки должны быть приведены ссылки на используемые литературные источники.

Источники нумеруются для того, чтобы на них можно было ссылаться из текста пояснительной записки.

**Например:**

1. Internet шаг за шагом [Электронный ресурс] : учебник. - СПб. : ПитерКом, 1997. - Электрон. дан. и программа. - 1 электрон. опт. диск (CD-ROM) + прил. (127 с.).

2. Александр и Наполеон. История двух императоров [Электронный ресурс] / Музей-панорама «Бородинская битва», Интерсофт. – Электрон. дан. – М. : Интерсофт, 1997. – 1 электрон. опт. диск (CD-ROM) : зв., цв. 12 см. – Систем. требования : ПК с процессором 486 DX2-66 ; 8 Мб ОЗУ ; Microsof Windows 3.1 или Windows 95 ; 2-скоростной дисковод CD-ROM ; видеокарта SVGA 256 цв. ; зв. карта 16 бит стандарта MPC ; стереоколонки или наушники. – Загл. с этикетки диска.

3. Библиотеки вузов Восточной Сибири в региональном информационном пространстве : Материалы науч.-практ. конференции [Электронный ресурс] – Иркутск : Науч. б-ка Иркут. ун-та, 2002. –http:// www.library.isy.ru/nauka/konf.htm (28 окт. 2002).

4. **Быков, В. Н.** История России [Текст] : учеб. пособие для обучающихся всех специальностей / В. Н. Быков ; М-во образования Рос. Федерации, С.-Петерб. гос. лесотехн. акад. – СПб. : СПбЛТА, 2001. – 231 с.

5. **Даль, В. И.** Толковый словарь живого великорусского языка Владимира Даля [Электронный ресурс] : подгот. по 2-му печ. изд. 1880–1882 гг. – М. : АСТ [и др.], 1998. – Электрон. дан. – 1 электрон. опт. диск (CD-ROM) ; 12 см + рук. пользователя (8 с.) – (Электронная книга). – Систем. требования : IBMPC с процессором 486 ; ОЗУ 8 Мб ; операц. система Windows (3х, 95, NT) ; CD-ROM дисковод ; мышь. – Загл. с экрана.

6. ЕСКД. Правила выполнения чертежей и схем оптических изделий [Текст] : ГОСТ 19.404-79. Введ. 1981-01-01. – М. : Изд-во стандартов, 1981.

7. **Киселев, В. В.** Анализ научного потенциала [Текст] / В. В. Киселев, Т. Е. Кузнецова, З. З. Кузнецов. – М. : Наука, 1991. – 126 с.

8. **Коломиец, Л.** Устойчивое развитие : Миф или реальность? [Электронный ресурс] / Л. Ко-ломиец. – Режим доступа : http://ust-razvitie.narod.ru/.

9. **Корнелиус, Х.** Выиграть может каждый : Как разрешать конфликты [Текст] : пер. П. Е. Патру-шева / Х. Корнелиус, З. Фэйр. – М. : Стрингер, 1992. – 116 с.

10.**Мудрик, А. В.** Воспитание в контексте социализации / А. В. Мудрик [Электронный ресурс] / Под патронажем Российской академии образования, ГНПБ им. К. Д. Ушинского // Образование : исследовано в мире. – М. : OIM.RU, 2000-2001. – Режим доступа : World Wide Web. URL : http://www.oim.ru. - 25.09.2000.

11.О местном референдуме в Хабаровском крае [Электронный ресурс] : Закон от 27.11.2002 г. № 74 // Справочно-правовая система «Гарант» ; НПП «Гарант-Сервис». – Послед. обновление 23.03.2003 г.

12.О ратификации консульского договора между Российской Федерацией и Китайской Народной Республикой [Электронный ресурс] : Федер. закон от 19.02.2003 г. № 31-ФЗ.

13.Образование : исследовано в мире [Электронный ресурс] : Международный научный педагогический Интернет-журнал с библиотекой-депозитарием / Под патронажем Российской академии образования, ГНПБ им. К. Д. Ушинского. – М. : OIM.RU, 2000-2001. – Режим доступа : World Wide Web. URL : http://www.oim.ru. - 10.02.2001.

14.Официальный сайт Президента Российской Федерации [Электронный ресурс] / Администт-рация Президента РФ. – Москва, 2001. – Режим доступа : www.president.kremlin.ru.

15.Российская Федерация. Законы. О воинской обязанности и военной службе [Текст] : Федер. закон. – М. : Ось-89, [2001?]. – 46, [1] с. – (Актуальный закон).

16.Российская Федерация. Конституция (1993). Конституция Российской Федерации [Текст] : офиц. текст. – М. : Маркетинг, 2001. – 39 с.

17. **Савинова, Ф.** Экологические проблемы и здоровье населения. 1989-1999 гг. [Электронный ре-сурс] / Ф. Савинова // Мир и безопасность. – 2000. – № 3. – Режим доступа : www.secur.ru/vitmib13.htm.

18.**Семенов, В. В.** Философия : итог тысячелетий. Философская психология [Текст] / В. В. Семенов ; Рос. акад. наук, Пущин. науч. центр, Ин-т биофизики клетки, Акад. проблем сохранения жизни. – Пущино : ПНЦ РАН, 2000. – 64 с.

19.Теория зарубежной судебной медицины [Текст] : учеб. пособие / В. Н. Алисиевич [и др.]. – М. : Изд-во МГУ, 1990. – 40 с.

20.**Фаронов, В. В.** Turbo Pascal [Текст] : учеб. пособие / В. В. Фаронов. – СПб. : Питер, 2006. – 366 с.

21.Художественная энциклопедия зарубежного классического искусства [Электронный ресурс]. – М. : Большая Рос. энцикл. [и др.], 1996. – Электрон. текстовые, граф., зв. дан. и прикладная прогр. (546 Мб). – 1 электрон. опт. диск (CD-ROM). – (Интерактивный мир). – Систем. требования : ПК 486 или выше ; 8 Мб ОЗУ ; Windows 3.1 или Windows 95 ; SVGA 32768 и более цв. ; 640×480 ; 4х CD-ROM дисковод ; 16-бит. зв. карта ; мышь. – Загл. с экрана.

22.**Цветков, В. Я.** Компьютерная графика : рабочая программа [Электронный ресурс] : для обучающихся заоч. формы обучения геодез. и др. специальностей / В. Я. Цветков. – М. : МИИГАиК, 1999. – Электрон. дан. и прогр. – 1 дискета. – Систем. требования : IBMPC, Windows 95, Word 6.0. – Загл. с экрана. – № гос. регистрации 0329900020.

23.**Яблоков, А. В.** Управление охраной природы – проблемы и решения [Электронный ресурс] / А. В. Яблоков. –: http://aeli.altai.ru/conferenc/1999/turina.html

## 4 Оформление пояснительной записки

Оформление пояснительной записки выполняется согласно общим методическим требованиям, принятым в образовательной организации (см. "Методические указания по выполнению курсовой работы по дисциплинам направления Информатика и ВТ").

## 5 ТЕМЫ КУРСОВЫХ РАБОТ

1. Разработать программу перестановки элементов матрицы блоками. Дана действительная квадратная матрица порядка 2n. Получить новую матрицу, переставляя ее блоки размером n×n:

а) крест-накрест;

б) по часовой стрелке (левый верхний блок становится правым верхним, правый верхний – правым нижним и т.д.).

2. Составить программу транспонирования целочисленной матрицы.

3. Составить программу заполнения квадратной матрицы натуральными числами «по спирали». Дана квадратная матрица порядка n. Заполнить ее натуральными числами 1, 2, 3, … $n^2$, записывая их в нее «по спирали». Например, для n = 5 получаем следующую матрицу:

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| 16 | 17 | 18 | 19 | 6 |
| 15 | 24 | 25 | 20 | 7 |
| 14 | 23 | 22 | 21 | 8 |
| 13 | 12 | 11 | 10 | 9 |

4. Разработать программу «Наибольший элемент». Дана действительная квадратная матрица порядка N (N – нечетное), все элементы которой различны. Найти наибольший элемент среди стоящих на главной и побочной диагоналях и поменять его местами с элементом, стоящим на пересечении этих диагоналей.

5. Разработать проект «Методы сортировки». Проект позволяет сортировать заданный линейный массив целых чисел различными методами, например, методом линейной сортировки, пузырька, Шелла и др. Предусмотреть использование не менее трех методов.

6. Разработать программу «Седловые точки». Для заданной матрицы размером (N×M) опре-делить индексы всех ее седловых точек. Элемент матрицы называется седловой точкой, если он является наименьшим в своей строке и одновременно наибольшим в своем столбце или, наоборот, является наибольшим в своей строке и наименьшим в своем столбце.

7. Составить программу «Магический квадрат». Программа проверяет, образуют ли элементы двумерного массива магический квадрат. В магическом квадрате – суммы чисел по всем верти-калям, всем горизонталям и двум диагоналям одинаковы.

8. Разработать программу «Перестановка». Дана вещественная матрица размером (N×M). Переставляя ее строки и столбцы, добиться того, чтобы наибольший элемент (или один из них) оказался в левом верхнем углу.

9. Разработать программу «Определение маршрута». В таблице размером (N×N), где N <= 20, клетки заполнены цифрами случайным образом. Найти маршрут из клетки (1, 1) в клетку (N, N), удовлетворяющий следующим условиям:

1) любые две последовательные клетки в маршруте имеют общую сторону;

2) количество клеток маршрута минимально;

3) сумма цифр в клетках маршрута максимальна.

10.Разработать проект «Арифметические операции над матрицами». Программа должна обеспечивать выполнение операций сложения, вычитания и умножения над матрицами целых чисел. Выбор выполняемой операции осуществляет пользователь.

11.Разработать проект «Сортировка строкового массива», Программа должна сортировать строковый массив (например, содержащий компьютерные термины) по алфавиту. Обеспечить сортировку внутри группы строк, начинающихся на одну и ту же букву (например, строка, содержащая слово ПРИНТЕР должна предшествовать строке, содержащей слово ПРОГРАММА).

12.Разработать программу «Удаление цепочки четных элементов массива». Из массива удалить самую длинную цепочку четных элементов.

*Пример:* из массива A[8]: 4 1 4 2 1 2 4 6 должен получиться массив A[5]: 4 1 4 2 1 (самая длинная цепочка четных чисел включает элементы с 6 по 8: 2 4 6).

13.Разработать программу «Удаление повторяющихся элементов». Из массива A удалить те элементы, которые встречаются и в массиве A и в массиве B по крайней мере по 2 раза.

*Пример:*

массив A[8]: 3 3 4 5 2 3 5 9;

массив B[7]: 1 2 3 4 5 2 5.

По 2 раза в обоих массивах встречается только элемент, равный 5.

Массив A после удаления примет вид: A[6]: 3 3 4 2 3 9.

14.Разработать программу «Поиск». Программа должна осуществлять поиск заданных слов в текстовом файле. Слова последовательно вводятся с клавиатуры. Для каждого слова должно определяться количество вхождений и номера строк текста. Если указанное слово в файле отсутствует, то программа должна выводить соответствующее сообщение.

15.Разработать программу «Синонимы». Даны 2 текстовых файла f1 и f2. Файл f1 содержит произвольный текст. Слова в тексте разделены пробелами и знаками препинания. Файл f2 содержит не более 30 слов, которые разделены запятыми. Эти слова образуют пары: каждое второе является синонимом первого. По возможности заменить в файле f1 слова их синонимами из файла f2. Результат поместить в новый файл.

16.Разработать программу «Сортировка по алфавиту». Дан текстовый файл. Переписать в алфавитном порядке все слова из заданного файла, имеющие длину n, в другой файл.

17.Разработать программу «Подсчет слов». Дан файл, содержащий текст на русском языке. Подсчитать количество слов, начинающихся и заканчивающихся на одну и ту же букву, и выдать эти буквы с указанием соответствующего количества слов.

18.Разработать программу «Список слов». Файл содержит текст на русском языке. Составить в алфавитном порядке список всех слов, встречающихся в тексте, и количество этих слов.

19.Разработать программу «Общие слова». Дан файл, содержащий текст на русском языке. Найти слова, встречающиеся в каждом предложении, или сообщить, что таких слов нет.

20.Разработать программу «Удаление повторяющихся слов». Дан файл, содержащий текст на русском языке. В предложениях некоторые из слов записаны подряд. Получить в новом файле отредактированный текст, в котором удалены повторные вхождения слов в предложение.

21.Написать программу «Шифр Цезаря». Программа позволяет зашифровывать и расшифровывать с помощью «шифра Цезаря» сообщение, написанное на русском языке. Этот шифр реализует следующее преобразование текста: каждая буква исходного текста заменяется третьей после нее буквой в алфавите, который считается написанным по кругу.

22.Написать программу «Шифр перестановки». Программа позволяет зашифровывать и рас-шифровывать сообщение с помощью «шифра перестановки». Этот шифр меняет местами две со-седние буквы.

23.Разработать проект «Русско-английский и англо-русский словарь». Программа обеспечи-вает перевод слов, хранящихся в файле данных или вводимых с клавиатуры.

24.Разработать программу «Удаление пробелов». Дан текстовый файл. Переписать его текст в новый файл таким образом, чтобы каждое предложение начиналось с новой строки и между словами осталось только по одному пробелу.

25.Разработать программу «Подсчет ключевых слов и идентификаторов». Текст программы на Паскале хранится в файле на диске. Составить программу обработки текста программы:

1) подсчитать, какие ключевые слова Паскаля и в каком количестве использованы в обрабаты-ваемом тексте;

2) составить перечень имен простых переменных, используемых в левой части оператора при-сваивания.

26.Разработать программу «Обработка текста программы». Текст программы на Паскале хранится в файле на диске. Составить программу обработки текста программы:

1) определить максимальную степень вложенности циклов в программе;

2) определить общее количество строк и количество символов, отличных от пробела;

3) удалить из текста программы все комментарии.

27.Разработать программу «Редактирование текста программы». Текст программы на Паскале хранится в файле на диске. Составить программу обработки текста программы:

1) первые буквы служебных слов сделать заглавными;

2) текст комментария заменить на номер комментария по порядку.

28.Разработать программу «Арифметическое выражение». Программа должна анализировать правильность записи арифметического выражения с точки зрения синтаксиса Паскаля. Арифме-тическое выражение задается строковой переменной и вводится с клавиатуры компьютера.

29.Разработать программу «Печать текста программы». Текст программы на Паскале хра-нится в файле на диске. Распечатать на экране текст программы таким образом, чтобы в каждой строке размещался только один оператор. Организовать смещение операторов относительно операторных скобок, как это принято в Паскале.

30.Разработать программу «Архивация». Программа должна позволять сжимать текстовую информацию, а затем преобразовывать сжатую информацию в исходное состояние. В программе необходимо предусмотреть два варианта. Для хранения текста в сжатом виде найти часто повто-ряющиеся последовательности из двух букв и заменить их кодом. В качестве кода использовать символы, не встречающиеся в тексте. Составить таблицу кодов. В заданном тексте найти слова, которые встречаются более трех раз, закодировать их и сжать текст, заменив слова кодами. Соста-вить таблицу кодов.

31.Разработать программу «Выравнивание». Исходная информация: текст, записанный в текстовом файле. Программа выводит этот текст с выравниванием по краям. Текст выводится без переносов слов. Параметры абзаца задаются в диалоговом режиме.

32. Разработать программу «Поиск операторов». Текст программы на Паскале хранится в файле на диске. Программа должна определять и выводить на экран операторы, которые изменяют значения заданных переменных (их имена вводятся) и номера строк программы, где они находятся.

33.Составить программу «Идентификаторы». Программа должна определять наличие неопи-санных идентификаторов в тексте программы на Паскале. Текст программы хранится в файле на диске.

34. Разработать приложение справочной службы кинотеатра. Программа должна обеспечивать:

- ввод и корректировку информации о забронированных билетах на конкретный сеанс;
- вывод плана зрительного зала с указанием свободных и купленных мест.

35.Разработать приложение справочной службы по аптекам города. Программа должна обеспечивать:

- вывод информации о наличии запрашиваемого лекарства в той или иной аптеке;
- поиск аптеки, в которой запрашиваемое лекарство продается по самой низкой цене.

36.Разработать приложение справочной службы железнодорожного вокзала. Программа должна выдавать справки о наличии билетов в спальные, купейные и плацкартные вагоны на все рейсы текущего месяца. Предусмотрите удобный интерфейс для пользователя.

37.Разработать приложение «Помощник экзаменатора». Экзаменационные вопросы и ответы к ним хранятся в файлах на диске. Каждый вопрос имеет балл сложности. Необходимо подобрать пять вопросов из разных разделов курса, имеющих в сумме балл сложности N, и вывести их на экран. Предусмотреть тренировочный режим работы, когда возможен вывод ответов на представ-ленные вопросы. Доступ к тренировочному режиму работы предоставляется по паролю.

38.Разработать приложение «Тестирование». Программа должна обеспечивать проведение тестирования по одному из разделов курса «Программирование». Выбор правильного ответа осуществляется при помощи переключателя. За каждый правильный ответ начисляется один балл. В конце теста выводятся его результаты. Необходимо предусмотреть тренировочный режим работы, когда возможен вывод ответов на представленные вопросы. Доступ к тренировочному режиму работы предоставляется по паролю.

39.Разработать приложение, имитирующее простейший органайзер.

40.Создать программу ведения базы данных личной библиотеки. Программа должна обеспе-чивать:

- ввод и корректировку информации о новых книгах;
- поиск информации о книгах определенного автора;
- поиск информации о книгах определенного жанра.

41.Создать программу ведения базы данных личной видеотеки. Программа должна обеспе-чивать ввод и корректировку информации о новых дисках (своих и взятых на время у друзей), а также выдавать информацию по запросам:

- имеется ли в наличии указанный диск, и если нет, то кому он отдан;
- имеются ли диски, взятые у друзей и которые надо отдать на этой неделе;
- выдать список дисков с видеофильмами заданного жанра.

42.Разработать приложение «Телефонный справочник».

43.Разработать приложение «Записная книжка». Программа должна обеспечивать ввод и корректировку информации, а по запросу поиск следующей информации:

- номер телефона указанного лица и ФИО по номеру телефона;
- почтовый адрес и адрес электронной почты указанного лица;
- ФИО лиц, чьи даты рождения приходятся на указанный месяц (неделю).

44.Разработать приложение «Склад». Программа должна обеспечивать ввод и корректировку информации и выдавать информацию по следующим запросам:

- имеется ли в наличии указанный товар и в каком количестве;
- кому, на какую сумму и какой товар был отпущен в заданный день;
- какова суммарная стоимость товаров на складе в отчетный день.

45.Разработать программу ведения базы данных футбольной команды университета. В БД фиксируются дата игры, результат, название команды противника, ФИО игроков, забивших гол. Программа должна выдавать информацию по следующим запросам:

- ФИО наиболее результативного игрока за отчетный период;
- информацию об игре с наихудшим результатом;
- количество игр за отчетный период, сыгранных с указанной командой противника.

46.Создать программу ведения базы данных «Сотрудники» научного учреждения «Прогресс». В базе данных содержится список сотрудников: фамилия, должность, подразделение, зарплата за один час работы. Количество отработанных часов для каждого сотрудника вводится с клавиатуры компьютера.

Ведение базы данных включает в себя следующие пункты:

а) ввод информации о сотрудниках и запись ее на диск;

б) удаление ненужной информации с файлов на диске;

в) корректирование записей базы данных;

г) вывод расчетной ведомости для каждого подразделения.

Расчетная ведомость имеет вид:

| № п/п | Фамилия | Начислено | Подоходный налог | Отчисления в пенсионный фонд | К выплате |
|---|---|---|---|---|---|

47.Создать программу ведения базы данных торговой фирмы. Программа включает в себя:

- формирование и корректирование файлов данных;
- расчет комиссионного вознаграждения сотрудников фирмы.

Файл данных о продавце включает его имя и фамилию, табельный номер, дату поступления на работу. Торговая фирма выплачивает продавцам комиссионное вознаграждение в размере 5 %, если товара продано на сумму менее 1000 долл. в день, и 6 %, если выручка составляет 1000 долл./день и выше. Продавцы, проработавшие в фирме более 10 лет, получают комиссионные на 1 % больше. Сумма выручки за день для каждого продавца вводится с клавиатуры ЭВМ. Организуйте вывод общих итогов по сумме выручки и сумме комиссионного вознаграждения за месяц.

48.Разработать справочную систему по стандартным функциям языка Турбо Паскаль.

49.Разработать справочную систему по операторам языка Турбо Паскаль.

50.Разработать программу, которая определяет «водящего» в детской игре. Водящий опреде-ляется с помощью считалки следующим образом. Все играющие встают в круг и начинают «считаться». Каждый раз тот, на ком закончилась считалка, выбывает из круга. Водит оставшийся. Исходное количество играющих n. Количество слов считалки m. Используйте кольцевой список.

51.Построить имитационную модель бензоколонки. На бензоколонке K стоек (1 стойка может обслуживать 1 автомобиль), каждый автомобиль обслуживается S сек. Интервал между моментами прибытия на бензоколонку автомобилей является случайной величиной, распределенной по зако-ну P(x). Если все стойки заняты, автомобиль становится в очередь. Для заданных P(x) и S опреде-лить возможно меньшее значение K для того, чтобы очередь не удлинялась.

52.Написать подпрограмму–функцию Form(S, X), где S – строка, X – вещественная пере-менная. В строке записано арифметическое выражение, содержащее переменную X, константы (целые или вещественные), операции +, -, *, /. Порядок операций определен скобками. Подпрограмма–функция возвращает значение арифметического выражения при заданном значении X.

53.Написать подпрограмму–функцию Form(S, X, Y), где S – строка, X и Y – вещественные переменные. В строке записано арифметическое выражение, содержащее переменные X и Y, константы (целые или вещественные), операции +, -, *, /. Порядок операций определен скобками. Подпрограмма–функция возвращает значение арифметического выражения при заданных значе-ниях X и Y.

54.Разработать программу решения задачи. Задано выражение в постфиксной форме (обратная польская запись). Вычислить значение этого выражения для заданных значений входящих в него переменных.

55.Составить программу решения “задачи коммивояжера”. Необходимо определить мини-мальную стоимость проезда коммивояжера по N городам с возвращением в исходную точку. Каж-дый город входит в маршрут только один раз. Предположить, что стоимость проезда из города i в город j такая же, как и из j в i.

56.Разработать программу составления списка заданий для параллельных процессоров. Три одинаковых центральных процессора могут выполнять M заданий. Каждое задание может быть выполнено на любом процессоре, и если задание загружено в процессор, оно находится в нем до полного завершения (т.е. задания не могут прерываться или разделяться между двумя или более процессорами). При i = 1, ..., M задание i требует времени ti для его выполнения. Для любого порядка заданий следующее задание из списка выполняется на первом освободившемся процес-соре. Определить оптимальный порядок заданий, т.е. такой, который дает возможность завершить все задания в кратчайшее время.

57.Разработать программу решения задач по работе с мультисписками. Даны две разрежен-ные матрицы, хранящиеся в виде мультисписков. Напишите:

1) процедуру получения третьего мультисписка, являющегося матрицей–суммой первых двух;

2) процедуру удаления N-й строки матрицы.

58.Разработать процедуру исключения вершины из двоичного дерева.

59.Написать программу, удаляющую из матрицы строку и столбец, содержащие наиболь-ший элемент. Матрица является разряженной и хранится в виде мультисписков.

60.Написать программу «Обратная польская запись». Программа должна представлять заданное арифметическое выражение в виде обратной польской записи и вычислять его значение. Для решения задачи использовать динамическую структуру стек.

61.Написать программу «Шахматная позиция». Программа генерирует или считывает шахматную позицию и определяет, не находится ли один из королей под шахом и не является ли шах матом. В программе предусмотреть два варианта ввода исходных данных:

1) шахматная позиция генерируется с помощью датчиков случайных чисел;

2) шахматная позиция вводится с клавиатуры ЭВМ.

62.Разработать программу, моделирующую игру «Алчность». Игра имеет следующие правила. Перед Вами большое число ящиков с деньгами. Сумма денег в каждом ящике – случайная величина. Вы выбираете ящик, открываете его и берете деньги из ящика, или отказываетесь от них. Если Вы берете деньги, игра кончается. В противном случае Вы можете выбрать другой ящик. Эта процедура повторяется максимум до пяти ящиков (деньги из пятого ящика должны быть взяты, если он открыт).

63.Разработать программу моделирующую игру «Чет/Нечет». Два игрока, “нечетный” и “четный”, по очереди ставят единицы и нули в незанятые позиции поля N на N. Каждый из игроков может ставить 1 или 0 в произвольную свободную позицию, тем самым занимая ее. Игра продолжается до заполнения всех позиций. После этого суммируются числа вдоль каждой строки, каждого столбца и главных диагоналей. Число ODD нечетных сумм сравнивается с числом EVEN четных сумм. Если ODD > EVEN, выигрывает “нечетный”; если EVEN > ODD, выигрывает “четный”; если ODD = EVEN, результат считается ничейным. Если одним из игроков является ЭВМ, то постройте для нее выигрышную стратегию.

64.Разработать программу, моделирующую игру “Кости”. Играющий называет любое число в диапазоне от 2 до 12 и ставку, которую он делает в этот ход. Программа с помощью датчика случайных чисел дважды выбирает числа от 1 до 6 (“бросает кубик”, на гранях которого цифры от 1 до 6). Если сумма выпавших цифр меньше 7 и играющий задумал число меньшее 7, он выигры-вает сделанную ставку. Если сумма выпавших цифр больше 7 и играющий задумал число большее 7, он также выигрывает сделанную ставку. Если играющий угадал сумму цифр, он получает в четыре ра-за больше очков, чем сделанная ставка. Ставка проиграна, если не имеет места ни одна из описан-ных ситуаций. В начальный момент у играющего 100 очков. В программе должно присутствовать графическое изображение поверхности кубика при каждом ходе игрока.

65.Разработать программу, моделирующую игру “Морской бой”. На поле 10 на 10 позиций стоят невидимые вражеские корабли: 4 корабля по 1 клетке, 3 корабля по 2 клетки, 2 корабля по 3 клетки, 1 корабль в 4 клетки. Необходимо поразить каждую из клеток кораблей. Два игрока вводят позиции кораблей в виде цифр (1, 2, 3, 4) в соответствующие элементы матрицы, тем самым опре-деляя конфигурацию и положение кораблей. Игроки по очереди “наносят удары” по кораблям противника. Если позиция корабля указана верно, то она помечается крестиком на поле. Предусмотреть вариант игры, когда одним из играющих является ЭВМ.

66.Разработать программу, моделирующую игру “Сбей самолет”. По экрану летят вражеские самолеты. Цель играющего – сбить их. Пусковая установка находится в нижней строке экрана. Пусковую установку можно перемещать по строке вперед и назад.

67.Составить программу обучения работе с клавиатурой. Программа должна выдавать на экран буквы, цифры, слова и фразы, которые следует набрать на клавиатуре, и оценивать правильность и скорость набора. В программе надо предусмотреть три уровня подготовленности обучающегося.

68.Разработать программу, моделирующую игру “Автомобильные гонки”.

69.Разработать программу “Напоминание”. После того как пользователь введет время сигнала и текст, который должен выводиться на экран в заданное время как напоминание о наступлении какого-либо события,

окно программы должно исчезнуть с экрана. Текст-напоминание в указанное время должно выводиться поверх всех открытых окон.

70.Разработать программу «Будильник». После того как пользователь введет время сигнала и выберет для него музыкальный фрагмент как напоминание о наступлении какого-либо события, окно программы должно исчезнуть с экрана. Выбранный звуковой сигнал должен раздаться в заданное время. Предусмотрите возможность задания различных музыкальных фрагментов для различных событий.

71.Разработать программу «Результаты сессии». Программа выводит на экран результаты экзаменационной сессии группы в виде:

а) гистограммы;

б) круговой диаграммы;

в) графика.

Выбор формы представления результатов сессии производится пользователем.

72.Написать программу «Проверка памяти». Программу можно использовать для оценки способности игрока запоминать числа. Программа должна последовательно выводить ряд чисел, например, восемь, а испытуемый стараться их запомнить и потом ввести с клавиатуры. Время, в течение которого игрок видит число, ограничено одной секундой. Программа должна быть «интеллектуальной». Сначала она предлагает запоминать ряд одноразрядных чисел, потом двухразрядных, трехразрядных и т.д. Переход на следующий уровень сложности должен осуществляться, если испытуемый правильно выполнил задание. После окончания теста программа должна вывести результат испытания по каждой группе чисел.

73.Написать программу «Угадайка». Компьютер случайным образом загадывает число от 1 до 100. Задача пользователя – за минимальное количество попыток, но не более шести, угадать это число, предлагая компьютеру свои варианты, в ответ на которые программа указывает номер попытки и сообщение о том, больше загаданное число или меньше введенного с клавиатуры. При правильной стратегии (делении интервала чисел пополам) наверняка угадать число можно за семь попыток. Игра состоит из пяти партий. После последней партии выводятся результаты.

74.Разработать программу, реализующую игру угадывания слова по буквам. Компьютер задает первую и последнюю буквы слова, количество букв и предлагает отгадать остальные. Время на обдумывание надо ограничить. Предусмотреть реакцию программы на ошибочные действия играющего.

75.Написать программу составления кроссворда.

76.Написать программу, реализующую простой калькулятор.

77.Написать программу перевода чисел из одной системы счисления в другую. Систему счисления (десятичная, двоичная, восьмеричная или шестнадцатеричная) для задания и перевода числа выбирает пользователь.

78.Написать программу, позволяющую выполнять арифметические действия над двоичными, восьмеричными или шестнадцатеричными числами. Систему счисления выбирает пользователь.

79.Написать программу MP3 Player. Программа должна обеспечить выбор прослушиваемого файла, регулировку громкости звука, перемотку файла.

## 6 Литература

**Основная учебная**

1. **Синицына, Т.Г.** Введение в программирование на языке С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

2. **Синицына, Т.Г.** Основы объектно-ориентированного программирования в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

3. **Синицына, Т.Г.** Программирование в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

4. **Львович И.Я.** Основы информатики [Электронный ресурс]: учебное пособие/ Львович И.Я., Преображенский Ю.П., Ермолова В.В.— Электрон. текстовые данные.— Воронеж: Воронежский институт высоких технологий, 2014.— 339 c. http://www.iprbookshop.ru/23359.— ЭБС «IPRbooks»

5. Фарафонов А.С. Программирование на языке высокого уровня [Электронный ресурс]: методические указания к проведению лабораторных работ по курсу «Программирование»/ Фарафонов А.С.—

Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013.— 32 c.— http://www.iprbookshop.ru/22912.— ЭБС «IPRbooks»

**Дополнительная**

1. **Глазырина, И.Б.** Введение в программирование. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

2. **Глазырина, И.Б.** Основные типы данных в Турбо Паскале. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

3. **Глазырина, И.Б.** Модульное программирование.[Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

4. **Глазырина, И.Б.** Динамические структуры. [Электронный ресурс]: рабочий учебник/ Лабзина, Т.А. - 2011. - http://lib.muh.ru.

**Приложение 1 ФОРМА СПЕЦИФИКАЦИИ**

| Обозначение | Наименование | Примечание |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

**МЕТОДИЧЕСКОЕ ПОСОБИЕ**
**ПО КУРСОВОЙ РАБОТЕ**

**ПРОГРАММИРОВАНИЕ**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Т.А. Борисова
Оператор компьютерной верстки Ф.Р. Калимуллина

**4309.Курс.01;МПК.01;2**

**МЕТОДИЧЕСКОЕ ПОСОБИЕ**
**ПО КУРСОВОЙ РАБОТЕ**

# СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (КУРС 1)

**МОСКВА 2018**

Разработано И.А. Лёвиной
Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве учебного пособия
для обучающихся

# МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО КУРСОВОЙ РАБОТЕ

## СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (КУРС 1)

В Методическом пособии изложены цель и основные задачи курсовой работы, на решение которых ориентируются обучающиеся при выполнении курсовой работы. Приведены темы курсовых работ и примерные дидактические планы по каждой теме, а также список рекомендуемой литературы.

Методическое пособие предназначено для обучающихся по направлению 09.03.01 «Информатика и вычислительные техника», руководителей курсовых работ по дисциплине «Современные информационные технологии (курс 1) », а также для организаторов учебного процесса.

# МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО КУРСОВОЙ РАБОТЕ «СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (КУРС 1)»

## 1 Общие положения

Методическое пособие адресовано обучающимся, выполняющим в процессе обучения курсовую работу по дисциплине 4309 «Современные информационные технологии (курс 1)», а также руководителям данной курсовой работы и организаторам учебного процесса.

Курсовая работа - самостоятельная разработка конкретной темы по изучаемой дисциплине с элементами научного анализа, предназначенная для формирования у обучающихся теоретических знаний и практических навыков, умений работать с литературой, анализировать источники, делать обстоятельные и обоснованные выводы.

Структура и содержание курсовой работы, порядок ее выполнения, оформления и аттестации определены учебно-методическим пособием «Курсовая работа. Порядок написания и оформления (9049.х1.01;4, 2014 г.)». В настоящем пособии изложены цель и основные задачи курсовой работы, темы курсовых работ и примерные дидактические планы по каждой теме.

## 2 Цель и задачи курсовой работы

*Целью курсовой работы* является, с одной стороны, систематизация, закрепление и расширение теоретических знаний по дисциплине, с другой, - приобретение и развитие обучающимся таких важных качеств, как:

– умение работать с литературой, анализировать источники по проблеме исследования, делать обстоятельные и обоснованные выводы;

– умение грамотно и логически обоснованно излагать свои мысли и идеи;

– умение четко формулировать и аргументировано обосновывать предложения и рекомендации по результатам выполненного исследования;

– способность к творческому и критическому мышлению;

– овладение аналитическими навыками, т.е. способностью искать и находить информацию, формулировать проверяемые гипотезы, выстраивать данные в определенном порядке и оценивать их и т.п.;

– овладение навыками самостоятельной исследовательской работы.

*Основными задачами* при выполнении курсовой работы являются:

1. Обоснование актуальности и значимости темы работы.

2. Исследование состояния и разработанности выбранной темы исследования.

3. Рассмотрение теоретических аспектов изучаемой проблемы, раскрытие основных понятий и терминов, относящихся к данной проблематике.

4. Сбор и анализ информации по проблеме с использованием современных средств получения, хранения и переработки информации.

5. Разработка практических рекомендаций и предложений по тематике курсовой работы.

6. Формирование навыков самостоятельной работы на всех этапах выполнения курсовой работы – от обоснования актуальности до формулировки выводов и рекомендаций.

## 3 Темы курсовых работ и примерные дидактические планы

Темы курсовых работ и примерные дидактические планы по темам приведены в таблице 1.

Дидактические планы названы примерными потому, что по усмотрению обучающегося и руководителя курсовой работы они могут быть расширены за счет включения в них дополнительных дидактических единиц.

Такое расширение должно быть обосновано необходимостью более полного раскрытия темы, а также эффективного достижения цели и задач, поставленных перед курсовой работой. При выполнении курсовой работы, в первую очередь, должны быть раскрыты дидактические единицы, приведенные в таблице 1.

Примерный дидактический план рекомендуется использовать обучающемуся также при составлении глоссария по теме курсовой работы.

Таблица 1. Темы курсовых работ и примерные дидактические планы по темам

| № | Тема курсовой работы | Примерный дидактический план по теме |
|---|---|---|
| 1 | Гипертекстовые технологии | **Общие понятия.** Общее понятие о гипертекстовой технологии. Преимущества гипертекста. Область применения гипертекстовых технологий.<br><br>**HTML - язык разметки гипертекста.** Задачи, решаемые при помощи HTML. Гипертекстовые ссылки. Состав HTML-документа.Современное развитие HTML.<br><br>**HTTP, URL, WWW, программы-клиенты и программы-серверы.** HTTP - протокол обмена гипертекстовой информацией. Принцип работы протокола. Основные методы доступа к данным. URL - универсальный идентификатор ресурсов, формат URL. World Wide Web (WWW). Понятие программ-клиентов и программ-серверов, использующих гипертекстовую модель |
| 2 | Моделирование случайных процессов на ЭВМ | **Общие понятия в технологии моделирования.** Суть компьютерного моделирования. Этапы, цели и средства компьютерного математического моделирования. Применение компьютерного моделирования в различных областях деятельности.<br>**Имитация базовой последовательности случайных чисел.** Случайные числа. Формирование возможных значений случайных величин с заданным законом распределения. Способы генерации случайных чисел: аппаратный, табличный, алгоритмический.<br><br>**Моделирование простейших случайных воздействий.** Моделирование случайных событий. Типичные случаи моделирования случайных событий: моделирование одного случайного события, моделирование полной группы попарно несовместимых событий, моделирование независимых совместных событий, моделирование совместных зависимых событий.<br><br>**Моделирование случайных векторов и процессов.** Моделирование в рамках многомерных распределений. Метод условных распределений. Метод Неймана. Моделирование случайных векторов в корреляционной теории. Метод линейного преобразования. Метод канонических преобразований. Метод разложения в ряд Фурье |
| 3 | Мультимедиа-технологии | **Обзор мультимедиа-технологии.** Характерные особенности мультимедиа-технологий. Линейная и нелинейная мультимедиа-технологии. Возможности мультимедиа-технологий.<br>**Применение мультимедиа-технологии.** Применение в Интернете. Компьютерная графика. Моделирование на компьютере.<br>**Мультимедиа-технологии в обучении.** Возможности использования мультимедиа-технологий в обучении. Преимущества применения мультимедиа-технологий в обучении. Обучающие мультимедиа-продукты. Дистанционное обучение с применением мультимедиа-технологий |
| 4 | Обзор существующих автоматизированных обучающих систем | **Общие понятия.** Понятие автоматизированной обучающей системы. Возможности индивидуализации обучения при помощи автоматизированных обучающих систем. Преимущества индивидуального обучения. Группы задач, решаемых в рамках автоматизированных обучающих систем.<br>**Типы автоматизированных обучающих систем.** Типы обучающих программ: тренировочные и контролирующие, наставнические, имитационные и моделирующие, развивающие игры. |

| № | Тема курсовой работы | Примерный дидактический план по теме |
|---|---|---|
| | | **Принципы построение автоматизированных обучающих систем.** Основные принципы программирования автоматизированных обучающих систем. Основные элементы автоматизированных обучающих систем. Требования, предъявляемые к автоматизированным обучающим системам.<br>**Модели обучения автоматизированных обучающих систем.** Модель программируемого обучения. Реализация моделей обучения на основе метода пакета прикладных программ. Реализация моделей обучения методом экспертных систем. Мультиагентный подход к реализации моделей обучения |
| 5 | Обзор языков программирования баз данных | **Язык QBE.** Основные возможности языка QBE. Средства генерации запросов MS Access. Использование QBE для создания запросов.<br>**Язык SQL.** Стандартизация SQL. Типы данных SQL. Возможности SQL.<br>**Transact-SQL.** Отличие Transact-SQL от языка SQL. Использование Transact-SQL в MS SQL Server. Типы данных в MS SQL Server |
| 6 | Обзор существующих экспертных систем | **Общие понятия.** Понятие экспертной системы. Особенности экспертных систем. Применение экспертных систем. Преимущества экспертных систем перед человеком-экспертом. Структура экспертной системы.<br>**Характеристики и базовые функции экспертных систем.** Отличие экспертных систем от других программ искусственного интеллекта. Функции экспертных систем: приобретение знаний, представление знаний, управление процессом поиска решения, разъяснение принятого решения.<br>**Модели представления знаний в экспертных системах.** Логическая модель. Модель, основанная на использовании правил. Модель, основанная на использовании фреймов. Модель семантической сети |
| 7 | Подход RAD (быстрой разработки приложений) | **Методология RAD.** Основные принципы методологии RAD. Ограничения методологии RAD. Применение технологии RAD.<br>**Фазы жизненного цикла в рамках методологии RAD.** Фаза анализа и планирования требований. Фаза проектирования. Фаза построения. Фаза внедрения.<br>**Визуальное программирование.** Визуальные инструменты RAD. Универсальные и специализированные средства визуального программирования. Типы языков визуального программирования.<br>**Событийное программирование.** Событийно-ориентированная логика приложения, построенного с помощью RAD. Прерывание, событие, сообщение. Программирование от приоритетов |
| 8 | Тестирование и отладка программного обеспечения | **Понятие тестирования.** Принципы тестирования программного обеспечения. Информационные потоки процесса тестирования. Этапы тестирования.<br>**Виды тестирования.** Тестирование элементов. Тестирование интеграции. Тестирование правильности. Системное тестирование. Тестирование восстановления. Тестирование безопасности. Тестирование производительности. Тестирование, основанное на ошибках. Тестирование, основанное на сценариях. Тестирование при экстремальном программировании.<br>**Стратегии тестирования.** Структурное тестирование (тестирование “белого ящика”). Функциональное тестирование (тестирование “черного ящика”).<br>**Отладка.** Понятие отладки**.** Синтаксические программные ошибки. Семантические (смысловые) программные ошибки. Спецификации программы. Трассировка программы. Экспериментальные методы отладки |

| № | Тема курсовой работы | Примерный дидактический план по теме |
|---|---|---|
| 9 | Области примене-ния искусственного интеллекта | **Понятие искусственного интеллекта.** Определение искусственного интеллекта. Основные подходы к разработке систем искусственного интеллекта. Основные понятия искусственного интеллекта: интеллект, алгоритм, интеллектуальная задача. Типы систем искусственного интеллекта. Связь науки об искусственном интеллекте с другими науками. Когнитология.<br>**Области применения искусственного интеллекта.** Восприятие и распознавание образов. Математика и автоматическое доказательство теорем. Игры. Понимание естественного языка. Машинное творчество. Интеллектуальные интерфейсы. Интеллектуальные роботы. Обучение и самообучение. Выявление и представление знаний экспертов в экспертных системах.<br>**Современный искусственный интеллект.** Существующие системы искусственного интеллекта: Перспективы развития |
| 10 | Объектные модели языков программи-рования | **Общие понятия.** Объектная модель. Основные положения объектной модели. Объектно-ориентированное проектирование. Объектно-ориентированный анализ. Объектно-ориентированное программирование. Преимущества объектной модели. Перспективы развития объектной модели.<br>**Главные элементы объектной модели.** Абстрагирование. Инкапсуляция. Модульность. Иерархия.<br>**Дополнительные элементы объектной модели.** Типизация**.** Параллелизм**.** Сохраняемость |
| 11 | Основания и история объектно-ориентированного подхода к программированию | **История развития объектно-ориентированного подхода.** Возникновение языков программирования. Классификация языков программирования. Структурное программирование. Объектно-ориентированное программирование.<br>**Объектно-ориентированный подход к программированию.** Технологии программирования. Сущность объектно-ориентированного подхода к программированию. Понятие класса. Понятие объекта.<br>**Концепции объектно-ориентированного подхода.** Наследование, полиморфизм, инкапсуляция, методы, свойства, модульность |
| 12 | Основы технологии имитационного моделирования | **Имитационное моделирование.** История развития систем имитационного моделирования. Применение компьютерного моделирования в различных областях деятельности. Понятие статистического эксперимента. Область применения и классификация имитационных моделей. Основа любой имитационной модели - описание динамики системы.<br>**Моделирование случайных факторов.** Методы генерации случайных чисел. Моделирование непрерывных случайных величин: метод последовательных сравнений, метод интерпретации.<br>**Применение сетевых моделей для описания параллельных процессов.** Сети Петри. Е-сети.<br>**Обработка и анализ результатов моделирования.** Оценка адекватности. Оценка устойчивости. Оценка чувствительности. Калибровка модели. Подбор параметров распределений. Критерии согласия. Оценка влияния и взаимосвязи факторов |
| 13 | Особенности языка Лисп | **Описание языка Лисп.** История развития. Применение. Основные понятия языка Лисп: атомы и списки. Синтаксис.<br>**Диалекты языка Лисп.** Маклисп. МуЛисп. Интерлисп. Франс Лисп. Зеталисп Лисп-машин. Коммон Лисп.<br>**Особенности языка Лисп.** Свойства, отличающие Лисп от других языков программирования. Эквивалентность представления программ и данных в языке. Рекурсия – основная управляющая структура языка. Структура данных «связанный список» |

| № | Тема курсовой работы | Примерный дидактический план по теме |
|---|---|---|
| 14 | Основные принципы системного подхода | **Основные понятия теории систем.** Основные понятия и характеристики общей теории систем: компоненты системы, границы системы, синергия, вход — преобразование — выход, цикл жизни, системообразующий элемент. Значение системного подхода.<br>**Описание компонентов и методики проведения системного анализа.** Основные компоненты системного анализа. Методика проведения системного анализа: описание системы, выявление и описание проблемы, выбор и реализация направления решения проблемы.<br>**Принципы системного подхода.** Принцип цели. Принцип двойственности. Принцип целостности. Принцип сложности. Принцип множественности. Принцип историзма. |
| 15 | Пролог - язык разработки систем, основанных на знаниях | **История возникновения языка Пролог.** Императивные и декларативные языки программирования. Логическое программирование. Этапы развития языка Пролог. Классическая логика и язык Пролог.<br>**Описание языка Пролог.** Термы и объекты. Факты. Запросы к базе данных. Унификация. Правила. Рекурсивные процедуры. Встроенные предикаты. Арифметические выражения. Основные разделы программ, написанных на языке Пролог.<br>**Современное использование Пролог.** Современные реализации языка. Применение. Перспективы развития |
| 16 | Языки имитацион-ного моделирования | **Общие понятия.** История развития систем имитационного моделирования. Понятие статистического эксперимента. Область применения и классификация имитационных моделей. Основа любой имитационной модели - описание динамики системы.<br>**Процессно-ориентированные языки.** Язык GPSS/360. Язык Q-GERT.<br>**Языки непрерывного имитационного моделирования.** Язык DYNAMO.<br>**Событийно-ориентированные языки.** GASP IV. SIMSCRIPT II<br>**Унифицированный язык моделирования UML.** Назначение. Модели и их представление. Моделирование использования систем. Моделирование структуры систем. Моделирование поведения систем. |
| 17 | Программы для офисной автоматизации | **Электронный офис.** Информационная технология автоматизированного офиса. Развитие офисной автоматизации: от традиционного офиса к производственному и электронному. Основные компоненты электронного офиса.<br>**Электронный документооборот.** Назначение систем управления электронными документами. Подсистемы автоматизации документооборота. |
| | | **Автоматизация ввода информации в компьютер.** Сканеры для ввода текстов и иллюстраций. Специальные типы сканеров: сканеры форм, штрих-сканеры. Программы распознавания текстов |
| 18 | Пакеты прикладных программ для бухгалтерского учета | **Прикладное программное обеспечение.** Понятие, назначение и состав прикладного программного обеспечения. Особенности интегрированных пакетов прикладных программ. Профессиональные пакеты прикладных программ.<br>**Теоретические аспекты бухгалтерских и аналитических программ.** Причины введения автоматизированного учета. Этапы автоматизации бухгалтерского учета в России. Характеристика автоматизированных систем бухгалтерского учета. Классификация бухгалтерских программ. Критерии выбора бухгалтерской программы<br>**Наиболее распространенные системы автоматизации бухгалтерского учета (САБУ).** САБУ 1С. САБУ фирмы "Омега". САБУ фирмы "АйТи". САБУ фирмы "Атлант-Информ". Другие системы автоматизации бухгалтерского учета |
| 19 | Перспективы управления распределенной информацией | **Принципы управления распределенной информацией.** Понятие распределенной базы данных. Управление распределенной информацией. Технологии распределенной обработки данных.<br>**Модели распределенных баз данных.** Однородные и неоднородные системы. Методы построения распределенных баз данных "сверху вниз" и "снизу вверх.<br>**Технологии распределенной обработки информации. Технологии клиент-** |

| № | Тема курсовой работы | Примерный дидактический план по теме |
|---|---|---|
| | | **сервер:** модель файлового сервера, модель удаленного доступа к данным, модель сервера базы данных, модель сервера приложений. Технологии объектного связывания данных. Технологии реплицирования данных |
| 20 | Программное обеспечение САПР | **Проектирование САПР.** Предпроектная стадия (НИР). Стадия эскизного проекта (ОКР). Стадия технического проекта. Стадия рабочего проекта. Стадия испытаний. Стадия опытной эксплуатации. Стадия внедрения.<br>**Принципы построения САПР.** Цели создания САПР. Состав САПР. Программное обеспечение САПР. Основные принципы построения САПР.<br>**Программное обеспечение САПР.** Прикладное программное обеспечение САПР. Системное программное обеспечение. Специфика информационного обеспечения САПР |
| 21 | Протокол ODBC | **ODBC. Общие понятия.** Определение ODBC**.**. Программное управление источниками данных ODBC. Причины использования ODBC.<br>**Технология ODBC.** Структура программного обеспечения ODBC. Диспетчер драйверов ODBC. Имена источников данных DSN.<br>**Протокол ODBC и его реализации**. Соответствие требованиям API ODBC. Соответствие требованиям SQL ODBC. Уровни функциональных возможностей ODBC API. Использование протокола ODBC в СУБД |
| 22 | Характеристики CASE-средств | **Общая характеристика и классификация CASE-средств.** Общая характеристика CASE-технологий. Применение. Компоненты Case-средств. Классификация CASE-средств по признакам. Требования к интегрированной CASE-технологии. Классификация CASE-средств по типам.<br>**Технология внедрения CASE-средств.** Определение потребностей в CASE-средствах. Определение критериев успешного внедрения CASE-средств. Разработка стратегии внедрения CASE-средств.<br>**Оценка и выбор CASE-средств.** Анализ рынка CASE-средств. Процесс оценки. Процесс выбора. Критерии оценки и выбора. Примеры Case-средств |
| 23 | Языки представления знаний | **Искусственный интеллект. Понятие искусственного интеллекта.** Определение искусственного интеллекта. Основные подходы к разработке систем искусственного интеллекта. Основные понятия искусственного интеллекта: интеллект, алгоритм, интеллектуальная задача.<br>**Модели представления знаний.** Типичные модели представления знаний: логическая модель, продукционная модель, модель, основанная на использовании фреймов, модель семантической сети.<br>**Характеристика языков представления знаний.** История развития. Требования к языкам представления знаний. Примеры языков представления знаний (KRL, RRL, ART и т.д.) |
| 24 | Современные системы программирования | **Общие понятия.** Понятие современной системы программирования. Системы визуального программирования.<br>**Назначение и составные элементы.** Обязательные компоненты современных систем программирования. Структура современной системы программирования.<br>**Примеры современных систем программирования.** Системы программирования компании Borland/Inprise. Системы программирования фирмы Microsoft |
| 25 | Области применения новых информационных технологий | **Обзор информационных технологий.** Теоретические и практические основы применения современных информационных технологий Классификация информационных технологий по области применения и по степени использования в них компьютеров.<br>**Информационных технологии в обучении.** Аспекты информатизации образования: методологический, экономический, технический, технологический, методический. Классификация обучающих систем. Системы дистанционного обучения.<br>**Информационных технологии в других областях деятельности.** Автоматизированные системы научных исследований, системы автоматизированного проектирования, Case-технологии, геоинформационные технологии и др. |

## 4 Литература


**Основная учебная**

1. **Корзаченко, О.В.** Информационные системы и технологии [Электронный ресурс]: монография/ Корзаченко О.В., Барбара А.Д., Косенко О.Н., Такаева М.А.— Электрон. текстовые данные.— М.: Перо, Центр научной мысли, 2012.— 140 с.— http://www.iprbookshop.ru/8983.— ЭБС «IPRbooks»
2. **Ботуз С.П.** Интеллектуальные интерактивные системы и технологии управления удаленным доступом. Методы и модели управления процессами защиты и сопровождения интеллектуальной собственности в сети Internet/Intranet [Электронный ресурс]: учебное пособие/ Ботуз С.П.— Электрон. текстовые данные.— М.: СОЛОН-ПРЕСС, 2014.— 340 c. http://www.iprbookshop.ru/26917.— ЭБС «IPRbooks»

**Дополнительная**

1. **Колдаев, В.Д.** Информационные системы и технологии. Часть 1 [Электронный ресурс]: монография/ В.Д. Колдаев [и др.].— Электрон. текстовые данные.— М.: Перо, Центр научной мысли, 2011.— 126 с.— http://www.iprbookshop.ru/8982..— ЭБС «IPRbooks»
2. **Стешин, А.И.** Информационные системы в организации [Электронный ресурс]: учебное пособие/ Стешин А.И.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2013.— 194 c.— http://www.iprbookshop.ru/16346.— ЭБС «IPRbooks»

**МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО КУРСОВОЙ РАБОТЕ**

**СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (КУРС 1)**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Г.Б. Казьмина
Оператор компьютерной верстки С.А. Кафтанников

**0619.01.03;МУАТ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**

**ПО ДИСЦИПЛИНЕ «ФИЗИКА»**
**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ**
**(РЕШЕНИЕ ЗАДАЧ)**

**МОСКВА 2018**

Разработано Яременко Ю.Г., к.ф.-м.н, доц.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для обучающихся

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**
**ПО ДИСЦИПЛИНЕ «ФИЗИКА»**
**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

# КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ (РЕШЕНИЕ ЗАДАЧ)

Методические указания (МУ) предназначены для обучающихся образовательной организации.

МУ направлены на овладение обучающимися системными знаниями по дисциплине по дисциплине «Физика».

**О Г Л А В Л Е Н И Е**

Стр.

**Введение**

***Цель*** методических указаний по проведению практических занятий – помочь обучающимся, изучающим курс «Физика», применять на практике полученные теоретические знания о физических телах и явлениях, рассмотреть физические законы и категории.

Методические материалы представляют собой комплекс практических занятий для аудиторной работы, а также указаний и разъяснений, позволяющих обучающему сформировать знания в области методологии физики.

Настоящие методические рекомендации по выполнению практических занятий по курсу «Физика» составлены на основе требований Федерального государственного образовательного стандарта высшего образования.

Основные ***задачи*** практических заданий:

- формирование у обучающихся естественнонаучного мировоззрения и развитие физического мышления;
- изучение основных физических явлений; овладение фундаментальными понятиями, законами и теориями классической и современной физики, а также методами физического исследования;
- овладение приемами и методами решения конкретных задач из различных областей физики;
- умение выделить конкретное физическое содержание в прикладных задачах будущей деятельности.

В практических работах оцениваются владение знаниями в области теории физики.

## УЧЕБНО-МЕТОДИЧЕСКОЕ, МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ

### а) Литература

**Основная учебная**

1 **Соболева В.В.** Общий курс физики [Электронный ресурс]: учебно-методическое пособие к решению задач и выполнению контрольных работ по физике/ Соболева В.В., Евсина Е.М.— Электрон. текстовые данные.— Астрахань: Астраханский инженерно-строительный институт, ЭБС АСВ, 2013.— 250 с.— http://www.iprbookshop.ru/17058.— ЭБС «IPRbooks»

2 **Дмитриева Е.И.** Физика для инженерных специальностей [Электронный ресурс]: учебное пособие/ Дмитриева Е.И.— Электрон. текстовые данные.— Саратов: Ай Пи Эр Медиа, 2012.— 142 с.— http://www.iprbookshop.ru/729.— ЭБС «IPRbooks»

**Дополнительная**

1 **Яременко, Ю.Г.** Основы молекулярной физики и термодинамики. [Электронный ресурс]: рабочий учебник/Яременко, Ю.Г. - 2010. - http://lib.muh.ru.

2 **Ваганова, Л.А**. Электричество и электромагнетизм. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2010. - http://lib.muh.ru.

3 **Бармасов, А.В.** Лабораторный практикум по дисциплине "Физика" [Электронный ресурс]: учебное пособие/ Бармасов А.В., Бармасова А.М., Белов М.М.— Электрон. текстовые данные.— СПб.: Российский государственный гидрометеорологический университет, 2006.— 119 с.— http://www.iprbookshop.ru/12492.— ЭБС «IPRbooks».

4 **Ваганова, Л.А.** Колебания и волны. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru

5 **Ваганова, Л.А.** Оптика. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru

6 **Ваганова, Л.А.** Основы квантовой механики и атомной физики. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru

7 **Ваганова, Л.А.** Элементы квантовых статистик и квантовой физики твердого тела. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru

8 **Ваганова, Л.А.** Механика. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru

9 **Ваганова, Л.А.** Физика атомного ядра и элементарных частиц. [Электронный ресурс]: рабочий учебник/ Ваганова, Л.А. - 2009. - http://lib.muh.ru
10 **Плешакова Е.О.** Физика. Механика [Электронный ресурс]: учебное пособие/ Плешакова Е.О.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2008.— 142 с.— http://www.iprbookshop.ru/11356.— ЭБС «IPRbooks»
11 **Мещерякова, Н.Е.** Физика. Оптика [Электронный ресурс]: учебное пособие/ Мещерякова Н.Е.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2009.— 70 с.— http://www.iprbookshop.ru/11358.— ЭБС «IPRbooks»
12 **Растова, Н.А**. Физика [Электронный ресурс]: учебное пособие/ Растова Н.А.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2009.— 42 с.— http://www.iprbookshop.ru/11357.— ЭБС «IPRbooks».

Ресурсы информационно-телекоммуникационной сети Интернет:

- http://www.edu.ru;
- http://ru.wikipedia.org - сайт электронной энциклопедии (см. статьи по физике);
- http://khodus.ucoz.ru/ - образовательный сайт по физике (см. видеофильмы по физике);
- http://faculty.ifmo.ru/butikov/Lectures/ - лекции по физике;
- http://sfiz.ru - электронный журнал «Современная физика».

– Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы.
- тренинговые и тестирующие программы.
- интеллектуальные роботизированные системы оценки качества выполненных работ.

– Информационные и роботизированные системы, программные комплексы, программное обеспечение для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ПО «Комбат»;
- ПО «ЛиК»;
- ПК «КОП»;
- ИР «Каскад».

**б) Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

## ПРАКТИЧЕСКАЯ РАБОТА № 1. ОСНОВЫ МОЛЕКУЛЯРНОЙ ФИЗИКИ И ТЕРМОДИНАМИКИ

***Цель занятия:*** изучить систему законов и принципов молекулярной физики и термодинамики.

***Основные понятия***

**Агрегатные состояния вещества –** состояния одного и того же вещества, переходы между которыми сопровождаются скачкообразным изменением ряда физических свойств

**Адиабатический процесс –** процесс, при котором отсутствует теплообмен ($\delta Q = 0$) между термодинамической системой и окружающей средой

**Вещество –** вид материи, совокупность дискретных образований, обладающих массой покоя (элементарные частицы, атомы, молекулы)

**Внутренняя энергия системы –** совокупность энергий всех видов молекулярных взаимодействий и энергии теплового движения молекул (атомов)

**Время релаксации термодинамической системы –** промежуток времени, в течение которого термодинамическая система приходит в равновесное состояние

**Закона Дюлонга и Пти –** молярная (атомная) теплоемкость химически простых тел в кристаллическом состоянии одинакова (равна 3R) и не зависит от температуры

**Закрытая термодинамическая система –** термодинамическая система, которая не может обмениваться с внешней средой ни энергией, ни веществом

**Идеальный газ –** модель в молекулярно-кинетической теории. Газ считается идеальным, если: 1) собственный объем молекул газа пренебрежимо мал по сравнению с объемом сосуда; 2) между молекулами газа отсутствуют силы взаимодействия; 3) столкновения молекул газа между собой и со стенками сосуда абсолютно упругие

**Капиллярные явления –** физические явления, обусловленные поверхностным натяжением на границе раздела несмешивающихся сред

**Критическое состояние термодинамической системы –** состояние термодинамической системы с критическими параметрами (рк, Vк, Тк). В критическом состоянии исчезают различия физических свойств жидкости и пара, находящихся в термодинамическом равновесии

**Круговой процесс (цикл) –** процесс, при котором система, пройдя через ряд состояний, возвращается в исходное

**Молекулярно-кинетическая теория строения вещества –** теория, в основе которой лежат три положения, подтвержденные экспериментально и теоретически: все тела состоят из мельчайших частиц – атомов, молекул или ионов; атомы, молекулы или ионы вещества всегда находятся в непрерывном хаотическом движении; между частицами любого вещества существуют силы взаимодействия

**Монокристаллы –** твердые тела, частицы которых образуют единую кристаллическую решетку

**Первое начало термодинамики –** теплота, сообщенная системе, идет на приращение внутренней энергии и совершение системой работы над внешними телами

**Политропный процесс –** тепловой процесс, в котором теплоемкость тела остается постоянной

**Равновесное состояние термодинамической системы –** состояние термодинамической системы, при котором все параметры имеют определенные значения и в котором система может оставаться сколько угодно долго

**Температура инверсии –** температура, при которой (для данного давления) происходит изменение знака эффекта Джоуля – Томсона

**Теорема Карно –** из всех периодически действующих тепловых машин, имеющих одинаковые температуры нагревателей (Т1) и холодильников (Т2), наибольшим КПД обладают обратимые машины; при этом КПД обратимых машин, работающих при одинаковых темпе-ратурах нагревателей (Т1) и холодильников (Т2), равны друг другу и не зависят от природы рабочего тела (тела, совершающего круговой процесс и обменивающегося энергией с другими телами), а определяются только температурами нагревателя и холодильника

**Термодинамика –** раздел теоретической физики, изучающий наиболее общие свойства систем, находящихся в состоянии равновесия, и процессы перехода между этими состояниями

**Термодинамическая система –** совокупность макроскопических тел, которые могут взаимодействовать между собой и с другими телами (внешней средой) – обмениваться с ними энергией и веществом

**Термодинамические параметры (параметры состояния) –** физические величины, служащие для характеристики состояния термодинамической системы

**Термодинамический процесс –** всякое изменение состояния рассматриваемой термодинамической системы, характеризующееся изменением ее термодинамических параметров

**Энтропия –** функция состояния термодинамической системы, изменение которой dS в равновесном процессе равно отношению количества теплоты dQ, сообщенного системе или отведенного от нее, к термодинамической температуре Т системы; мера вероятности пребывания системы в данном состоянии

**Эффект Джоуля – Томсона –** изменение температуры реального газа в результате его адиабатического расширения (адиабатического дросселирования)

**Явления переноса –** группа необратимых процессов, связанных с выравниванием неоднородностей плотности, температуры или скорости упорядоченного перемещения отдельных слоев вещества

***Описание работы***

Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

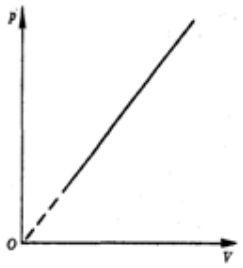
***Задание 1*** В двух сосудах одинакового объема находятся вода и ртуть. Сравните число атомов в этих жидкостях.

***Задание 2*** Сосуд объемом $V$ = 100 л разделен пополам полупроницаемой перегородкой. В начальный момент времени в одной половине сосуда находился водород, масса которого $m_1$= 2 г, а во второй – 1 моль азота. Определите давления, установившиеся по обе стороны перегородки, если она может пропускать водород. Температура в обеих половинах одинакова и постоянна: $t$ = 127 °С.

***Задание 3*** После опускания в воду, имеющую температуру $t_1$ = 10 °С, тела, нагретого до температуры $t_2$ = 100 °С, через некоторое время установилась общая температура $t_3$ = 40 °С. Какой станет температура воды $t$, если, не вынимая первого тела, в нее опустить еще одно такое же тело, нагретое тоже до температуры $t_2$ = 100 °С?

***Задание 4*** Пусть азот нагревается при постоянном давлении. Зная, что масса азота 280 г, количество затраченного тепла равно 600 Дж и $C_v$ равно 745 Дж/(кг К), определить повышение температуры азота.

***Задание 5*** На рисунке изображен график изменения состояния *идеального газа* в координатах $p$, $V$. Вычертите графики этого процесса в координатах $V$, $T$ и $p$, $T$.



**ПРАКТИЧЕСКАЯ РАБОТА № 2. Электричество и электромагнетизм**

***Цель занятия:*** изучить систему законов и принципов физики электричества и электромагнетизма.

***Основные понятия***

**Взаимная индукция –** явление возникновения ЭДС в одном из контуров при изменении силы тока в другом

**Вихревое электрическое поле –** электрическое поле с замкнутыми силовыми линиями, порождаемое переменным магнитным полем

**Диэлектрическая проницаемость среды** $\varepsilon$ **–** безразмерная величина, показывающая, во сколько раз поле ослабляется диэлектриком

**Магнитное поле –** одна из форм электромагнитного поля, создаваемого движущимися электрическими зарядами и спиновыми магнитными моментами атомных носителей магнетизма (электронов, протонов)

**Магнитный поток –** поток вектора магнитной индукции через какую-либо поверхность

**Напряжение –** физическая величина, определяемая работой, совершаемой суммарным полем электростатических (кулоновских) и сторонних сил при перемещении единичного положительного заряда на данном участке цепи: $U_{12} = \varphi_1 - \varphi_2 + \xi_{12}$

**Плазма –** сильно ионизованный газ, в котором концентрации положительных и отрицательных зарядов практически одинаковы

**Принцип суперпозиции электростатических полей –** напряженность $\vec{E}$ результирующего поля, создаваемого системой зарядов, равна геометрической сумме напряженностей полей, создаваемых в данной точке каждым из зарядов в отдельности

**Пробивное напряжение –** разность потенциалов между обкладками конденсатора, при которой происходит пробой – электрический разряд через слой диэлектрика в конденсаторе

**Самоиндукция –** явление возникновения электродвижущей силы (эдс индукции) в замкнутом проводящем контуре при изменении в нем силы тока

**Самостоятельный разряд –** разряд в газе, сохраняющийся после прекращения действия внешнего ионизатора

**Сверхпроводимость –** явление, наблюдаемое у некоторых веществ (сверхпроводников), состоящее в скачкообразном обращении в нуль электрического сопротивления постоянному току при охлаждении образца ниже определенной критической температуры $T_K$

**Сторонние силы –** силы неэлектростатического происхождения, действующие на заряды со стороны источников тока

**Ускорители заряженных частиц –** устройства, в которых под действием электрических и магнит-ных полей создаются и управляются пучки высокоэнергетичных заряженных частиц (электронов, протонов, мезонов и т.д.)

**Ферромагнетики –** сильномагнитные вещества, обладающие спонтанной намагниченностью, т.е. они намагничены даже при отсутствии внеш-него магнитного поля

**Электрический диполь –** система двух равных по модулю разноименных точечных зарядов (+$Q$, -$Q$), расстояние $l$ между которыми значительно меньше расстояния до рассматриваемых точек поля

**Электрический заряд –** величина, определяющая интенсивность электромагнитного взаимодействия заряженных частиц, источник электромагнитного поля

**Эквипотенциальные поверхности –** поверхности, во всех точках которых потенциал $\varphi$ имеет одно и то же значение

**Электрический ток –** любое упорядоченное (направленное) движение электрических зарядов; условно за направление электрического тока принимают направление движения положительного заряда

**Электродинамика –** раздел учения об электричестве, в котором рассматриваются явления и процессы, обусловленные движением электрических зарядов или макроскопических заряженных тел

**Электростатическая индукция –** явление перераспределения поверхностных зарядов на проводнике во внешнем электростатическом поле

**Электростатическое поле –** электрическое поле, которое создается неподвижными электрическими зарядами

***Описание работы***

Выполнить задания.

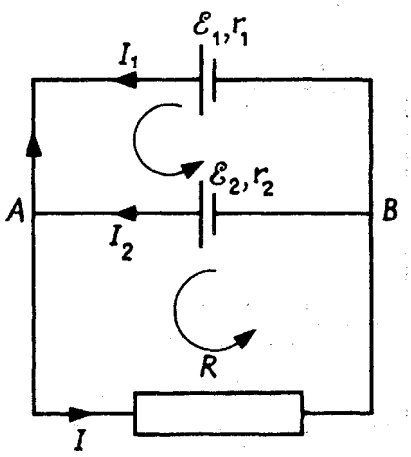
Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** Два заряженных шарика, подвешенные на нитях одинаковой длины, опускают в керосин, плотность которого 0,8 г/см$^3$. Какой должна быть плотность материала шариков, чтобы угол расхождения нитей в воздухе и керосине был один и тот же? Диэлектрическая проницаемость керосина ε=2.

***Задание 2.*** Расстояние между двумя точечными зарядами 2 нКл и -3 нКл, расположенными в вакууме, равно 20 см. Определите напряженность и потенциал поля, создаваемого этими зарядами в точке, удаленной от первого заряда на растояние 15 см и от второго заряда на расстояние 10 см.

***Задание 3.*** К двум последовательно соединенным сопротивлениям $R_1$ = 3 Ом и $R_2$ = 6 Ом параллельно подключены два последовательно соединенных конденсатора $C_1$ = 1 мкФ и $C_2$ = 2 мкФ. Данный участок подключен к источнику электрической энергии с ЭДС 12 В и внутренним сопротивлением 1 Ом. Найти ток в цепи и заряд накопленный каждым конденсатором.

***Задание 4.*** Рассчитать токи во всех участках цепи, изображенной на рисунке, если $\xi_1$=2 В, $\xi_2$=4 В, $r_1 = r_2$ = 2 Ом, $R$ = 9 Ом.




***Задание 5.*** Два прямолинейных длинных проводника, по которым текут токи $I_1$ = 10 А и $I_2$ = 20 А в противоположных направлениях, находятся на расстоянии 5 см друг от друга. Определите индукцию магнитного поля в точке $A$, расположенной на расстоянии 4 см от первого проводника и 3 см от второго.

***Задание 6.*** Катушка сопротивлением 50 Ом и индуктивностью $10^{-3}$ Гн находится в магнитном поле. При равномерном изменении магнитного поля поток через катушку возрос на $10^{-3}$ Вб и ток в катушке увеличился на 0,1 А. Какой заряд прошел за это время по катушке?

**ПРАКТИЧЕСКАЯ РАБОТА № 3. Колебания и волны**

***Цель занятия:*** изучить систему законов и принципов физики колебаний и волн.

***Основные понятия***

**Автоколебания –** незатухающие колебания, поддерживаемые в диссипативной системе за счет постоянного внешнего источника энергии, причем свойства этих колебаний определяются самой системой

**Бегущая волна –** волна, которая переносит в пространстве энергию

**Биения –** периодические изменения амплитуды колебания, возникающие при сложении двух одинаково направленных гармонических колебаний с близкими частотами

**Волновой процесс –** процесс распространения колебаний в сплошной среде

**Вынужденные механические колебания –** колебания, возникающие под действием внешней периодически изменяющейся силы

**Вынужденные электромагнитные колебания –** колебания, возникающие под действием внешней периодически изменяющейся ЭДС

**Интенсивность звука (или сила звука) –** величина, определяемая средней по времени энергией, переносимой звуковой волной в единицу времени сквозь единичную площадку, перпендикулярную направлению распространения волны

**Интерференция волн –** явление наложения в пространстве двух (или нескольких) когерентных волн, вследствие которого в разных его точках получается усиление или ослабление результирующей волны в зависимости от соотношения между фазами этих волн

**Математический маятник –** идеализированная система, состоящая из материальной точки массой $m$, подвешенной на нерастяжимой невесомой нити, и колеблющаяся под действием силы тяжести

**Период колебания –** промежуток времени $T$, через который повторяются определенные состояния системы, совершающей гармонические колебания, а фаза колебания получает приращение $2\pi$

**Поперечная волна –** упругая волна, при которой частицы среды колеблются в плоскостях, перпендикулярных направлению распространения волны

**Принцип суперпозиции (наложения) волн –** при распространении в линейной среде нескольких волн каждая из них распространяется так, как будто другие волны отсутствуют, а результирующее смещение частицы среды в любой момент времени равно геометрической сумме смещений, которые получают частицы, участвуя в каждом из слагающих волновых процессов

**Резонансная частота –** частота, при которой амплитуда $A$ смещения (заряда) вынужденных колебаний достигает максимума

**Свободные затухающие колебания –** колебания, амплитуды которых из-за потерь энергии реальной колебательной системой с течением времени уменьшаются

**Свободные (собственные) колебания –** колебания, которые совершаются за счет первоначально сообщенной энергии при последующем отсутствии внешних воздействий на колебательную систему

**Сложение колебаний –** нахождение закона результирующих колебаний системы, если эта система участвует в нескольких колебательных процессах

**Спектр колебания –** совокупность частот простых гармонических колебаний, в результате сложения которых может быть получено сложное колебание системы

**Фазовая скорость –** скорость перемещения фазы волны

**Физический маятник –** твердое тело, совершающее под действием силы тяжести колебания вокруг неподвижной горизонтальной оси, проходящей через точку $O$, не совпадающую с центром масс $C$ тела

**Частота колебаний –** число полных колебаний, совершаемых в единицу времени

**Электромагнитная волна –** переменное электромагнитное поле, распространяющееся в пространстве с конечной скоростью

***Описание работы***

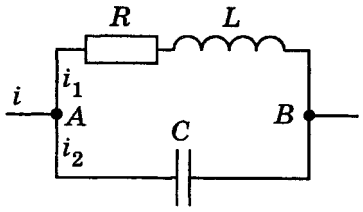
Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** Материальная точка массой 50 г колеблется по закону $x = 0{,}05\cos\pi(0{,}2t + 0{,}25)$. Напишите уравнения для скорости и ускорения этой точки. Найдите максимальную силу, действующую на точку, и полную энергию колеблющейся точки. Какова скорость точки при $t$ = 10 с?

***Задание 2.*** В цепь переменного тока стандартной частоты ( $\nu$ = 50 Гц) последовательно включены резистор сопротивлением $R$ = 21 Ом, катушка индуктивностью $L$ = 0,07 Гн и конденсатор емкостью $C$ = 82 мкФ. Определите индуктивное, емкостное и полное сопротивления цепи, а также сдвиг фаз между силой тока и напряжением. Какова активная мощность, если сила тока в цепи равна 2 А?

***Задание 3.*** В цепи (см. рис.) параметры $R$, $L$ и С известны. Напряжение между точками $A$ и $B$ равно $U$. Постройте векторную диаграмму сил токов в данной цепи и определите силу тока в неразветвленном участке цепи. Найдите сдвиг фаз между колебаниями силы тока и напряжения.




***Задание 4.*** Колебательный контур состоит из катушки индуктивности $4 \cdot 10^{-6}$ Гн и конденсатора, емкость которого можно изменять от 0,02 до 0,006 мкФ. Сопротивление контура ничтожно мало. На какой диапазон длин волн можно настроить колебательный контур?

**ПРАКТИЧЕСКАЯ РАБОТА № 4. Оптика**

---

***Цель занятия:*** изучить систему законов и принципов оптики.

***Основные понятия***

**Геометрическая оптика –** раздел оптики, в котором законы распространения света рассматриваются на основе представления о световых лучах

**Главная оптическая ось –** прямая, проходящая через центры кривизны поверхностей линзы

**Дисперсия света –** зависимость показателя преломления $n$ вещества от частоты $\nu$ (длины волны $\lambda$) света или зависимость фазовой скорости v световых волн от его частоты $\nu$

**Длина когерентности –** расстояние, при прохождении которого две или несколько волн утрачивают когерентность

**Закон независимости световых пучков –** эффект, производимый отдельным пучком не зависит от того, действуют ли одновременно остальные пучки или они устранены

**Закон отражения света –** отраженный луч лежит в одной плоскости с падающим лучом и перпендикуляром, проведенным к границе раздела двух сред в точке падения; угол $i_1'$ отражения равен углу $i_1$ падения: $i_1' = i_1$.

**Закон преломления света –** луч падающий, луч преломленный и перпендикуляр, проведенный к границе раздела в точке падения, лежат в одной плоскости; отношение синуса угла падения к синусу угла преломления есть величина постоянная для данных сред

**Закон прямолинейного распространения света –** свет в оптически однородной среде распространяется прямолинейно

**Линзы –** прозрачные тела, ограниченные двумя поверхностями (одна из них обычно сферическая, иногда цилиндрическая, а вторая – сферическая или плоская), преломляющими световые лучи, способные формировать оптические изображения предметов

**Молекулярное рассеяние –** рассеяние света в чистых средах, обусловленное флуктуациями плотности, анизотропии или концентрации

**Оптическая длина пути –** произведение геометрической длины пути световой волны *s* в данной среде на показатель *n* преломления этой среды

**Оптическая система –** комбинация собирающих и рассеивающих линз или линз и зеркал, в которых последовательно получаются изображения предмета

**Оптически активные вещества –** вещества, обладающие способностью вращать плоскость поляризации

**Оптический центр линзы –** точка, лежащая на главной оптической оси и обладающая тем свойством, что лучи проходят сквозь нее не преломляясь

**Плоское зеркало –** плоская поверхность, зеркально отражающая свет

**Принцип Гюйгенса – Френеля –** световая волна, возбуждаемая каким-либо источником *S*, может быть представлена как результат суперпозиции когерентных вторичных волн, «излучаемых» фиктивными источниками

**Принцип Ферма (принцип наименьшего времени) –** действительный путь распространения света (траектория светового луча) есть путь, для прохождения которого свету требуется минимальное время по сравнению с любым другим мыслимым путем между теми же точками

**Тонкая линза –** линза, толщина которой (расстояние между ограничивающими поверхностями) значительно меньше по сравнению с радиусами поверхностей, ограничивающих линзу

**Фокальные плоскости –** плоскости, проходящие через фокусы линзы перпендикулярно ее главной оптической оси.

**Фокус –** точка, в которой после преломления собираются все лучи, падающие на линзу параллельно главной оптической оси

**Фотометрия –** раздел оптики, занимающийся вопросами измерения интенсивности света и его источников

**Электронная оптика –** область физики и техники, в которой изучаются вопросы формирования, фокусировки и отклонения пучков заряженных частиц и получения с их помощью изображении под действием электрических и магнитных полей в вакууме

***Описание работы***

Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** Кажущаяся глубина водоема 3 м. Определите истинную глубину водоема $h_0$. Показатель преломления воды $N = 1{,}33$.

***Задание 2.*** Постройте изображение точечного источника света в собирающей линзе, если он находится на главной оптической оси между фокусом и двойным фокусным расстоянием.

***Задание 3.*** Постройте изображение точечного источника света в собирающей линзе, если он находится не на главной оптической оси между фокусом и двойным фокусным расстоянием.

***Задание 4.*** Найдите фокусное расстояние *f* и оптическую силу Ф собирающей линзы, если известно, что изображение предмета, помещенного на расстоянии 24 см от линзы, получается по другую сторону линзы на расстоянии 48 см от нее.

***Задание 5.*** На дифракционную решетку, имеющую 40 штрихов на 1 мм, падает монохроматический свет и образует полосу четвертого порядка под углом $\varphi = 6{,}4^{\circ}$. Чему равна длина волны падающего света.

**ПРАКТИЧЕСКАЯ РАБОТА № 5. Основы квантовой механики и атомной физики**

---

***Цель занятия:*** изучить систему законов и принципов квантовой механики и атомной физики.

***Основные понятия***

**Гипотеза де Бройля –** любой частице, обладающей импульсом, сопоставляют волновой процесс с длиной волны, определяемой по формуле де Бройля: $\lambda = h/p$, где $h$ – постоянная Планка, а $p$ – импульс частицы

**Закон Кирхгофа –** отношение спектральной плотности энергетической светимости к спектральной поглощательной способности не зависит от природы тела; оно является для всех тел универсальной функцией частоты (длины волны) и температуры

**Закон смещения Вина –** длина волны $\lambda_{max}$ , соответствующая максимальному значению спектральной плотности энергетической светимости $r_{\lambda T}$ черного тела, обратно пропорциональна его термодинамической температуре

**Закон Стефана – Больцмана –** энергетическая светимость черного тела пропорциональна четвертой степени его термодинамической температуры

**Магнитное спиновое квантовое число –** число $m_s = \pm 1/2$, определяющее проекцию спина на заданное направление

**Оптическая пирометрия –** методы измерения высоких температур, использующие зависимость спектральной плотности энергетической светимости или интегральной энергетической светимости тел от температуры

**Орбитальное квантовое число –** число $l$, определяющее момент импульса электрона в атоме

**Принцип детального равновесия –** при термодинамическом равновесии каждому процессу можно сопоставить обратный процесс, причем скорости их протекания одинаковы

**Принцип неразличимости тождественных частиц –** невозможно экспериментально различить тождественные частицы

**Принцип Паули –** системы фермионов встречаются в природе только в состояниях, описываемых антисимметричными волновыми функциями

**Принцип соответствия Бора –** законы квантовой механики должны при больших значениях квантовых чисел переходить в законы классической физики

**Соотношение неопределенностей –** произведение неопределенностей координаты и соответствующей ей проекции импульса не может быть меньше величины порядка $h$

**Состояние с инверсией населенностей –** чтобы среда усиливала падающее на нее излучение, необходимо создать неравновесное состояние системы, при котором число атомов в возбужденных состояниях было бы больше, чем их число в основном состоянии

**Спектральная плотность энергетической светимости (излучательности) тела –** количественная характеристика теплового излучения; мощность излучения с единицы площади поверхности тела в интервале частот единичной ширины

**Спектральная поглощательная способность –** характеристика способности тел поглощать падающее на них излучение; показывает, какая доля энергии, приносимой за единицу времени на единицу площади поверхности тела падающими на нее электромагнитными волнами с частотами от $\nu$ до $d\nu$ поглощается телом

**Универсальная функция Кирхгофа –** спектральная плотность энергетической светимости черного тела

**Фермионы –** частицы с полуцелым спином, которые описываются антисимметричными волновыми функциями и подчиняются статистике Ферми – Дирака

**Фотон –** элементарная частица, которая всегда (в любой среде!) движется со скоростью света и имеет массу покоя, равную нулю

**Фотоэлементы –** приемники излучения, работающие на основе фотоэффекта и преобразующие энергию излучения в электрическую

**Цветовая температура** – температура черного тела, при которой относительные распределения спектральной плотности яркости черного тела и рассматриваемого тела максимально близки в видимой области спектра

**Эффект Зеемана –** расщепление энергетических уровней в магнитном поле

**Эффект Комптона** – упругое рассеяние коротковолнового электромагнитного излучения (рентгеновского и γ-излучений) на свободных (или слабосвязанных) электронах вещества, сопровождающееся увеличением длины волны

**Эффект Штарка –** расщепление уровней энергии во внешнем электрическом поле

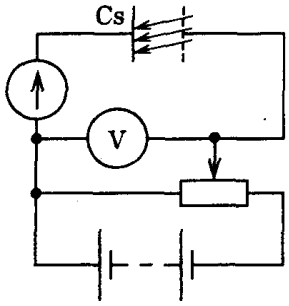
**Яркостная температура –** температура черного тела, при которой для определенной длины волны его спектральная плотность энергетической светимости равна спектральной плотности энергетической светимости исследуемого тела

***Описание работы***

Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** Определение зависимости запирающего напряжения фототока (напряжения, при котором фототок прекращается) от длины волны света, падающего на цезиевую пластину, выполняется по схеме, изображенной на рисунке. При освещении светом с длиной волны $\lambda_1$ = 0,40 мкм запирающее напряжение составляло $U_1$ = 1,19 *В* при $\lambda_2$ = 0,50 мкм $U_2$= 0,57 В. Определите по результатам этого опыта красную границу фотоэффекта для цезия и постоянную Планка.




***Задание 2.*** Определить энергию, импульс и массу фотона, длина волны которого соответствует рентгеновскому излучению $\lambda_1$ = 10 нм.

***Задание 3.*** Определите длину волны λ, испускаемую при переходе иона гелия из стационарного состояния с номером 4 в состояние с номером *2*.

***Задание 4.*** Пылинка массой $10^{-5}$ кг движется со скоростью 10 м/с. Проявляет ли она при своем движении волновые свойства?

**ПРАКТИЧЕСКАЯ РАБОТА № 6. Элементы квантовых статистик и квантовой физики твердого тела**

***Цель занятия:*** изучить систему законов и принципов квантовых статистик и квантовой физики твердого тела.

***Основные понятия***

**Вырожденная система частиц** – система, свойства которой существенным образом отличаются от свойств систем, подчиняющихся классической статистике

**Дырочная проводимость –** проводимость собственных полупроводников, обусловленная квазичастицами – дырками

**Квантовая статистика –** раздел статистической физики, исследующий системы, которые состоят из огромного числа частиц, подчиняющихся законам квантовой механики

**Контактная разность потенциалов –** разность потенциалов, возникающая при соприкосновении двух различных металлов

**Кристаллофосфоры –** твердые тела, представляющие собой эффективно люминесцирующие искусственно приготовленные кристаллы с чужеродными примесями

**Люминесценция –** неравновесное излучение, избыточное при данной температуре над тепловым излучением тела и имеющее длительность, большую периода световых колебаний

**Люминофоры –** вещества, способные под действием различного рода возбуждений светиться

**Макроскопическое квантование –** квантование величин, характеризующих макроскопические тела, размеры которых в $10^5$ раз больше атомных размеров

**Основная задача квантовой статистики –** определить средние значения величин, характеризующих рассматриваемую систему

**Полупроводники –** твердые тела, которые при T = 0 характеризуются полностью занятой электронами валентной зоной, отделенной от зоны проводимости сравнительно узкой ($\Delta E$ порядка 1 эВ) запрещенной зоной

**Правило Стокса –** длина волны люминесцентного излучения всегда больше длины волны света, возбудившего его

**Примесная проводимость –** проводимость полупроводников, обусловленная примесями

**Собственные полупроводники –** химически чистые полупроводники

**Температура вырождения –** температура, ниже которой отчетливо проявляются квантовые свойства идеального газа, обусловленные тождественностью частиц

**Термопары –** датчики температур, состоящие из двух соединенных между собой разнородных металлических проводников

**Уровень Ферми –** наивысший энергетический уровень, занятый электронами

**Фонон –** квант энергии звуковой волны

**Фотопроводимость полупроводников –** увеличение электропроводности полупроводников под действием электромагнитного излучения

**Экситоны –** квазичастицы, электрически нейтральные связанные состояния электрона и дырки, образующиеся в случае возбуждения с энергией, меньшей ширины запрещенной зоны

**Электронная проводимость –** проводимость собственных полупроводников, обусловленная электронами

**Электронно-дырочный переход –** граница соприкосновения двух полупроводников, один из ко-торых имеет электронную, а другой – дырочную проводимость

**Энергетический выход люминесценции –** отношение энергии, излученной люминофором при полном высвечивании, к энергии, поглощенной им

**Эффект Джозефсона –** протекание сверхпроводящего тока сквозь тонкий слой диэлектрика, разделяющий два сверхпроводника

**Эффективная масса –** величина, имеющая размерность массы и характеризующая динамические свойства квазичастиц – электронов проводимости и дырок

**Эффект Мейснера –** физическое явление, которое заключается в том, что при охлаждении сверхпроводника ниже критической температуры магнитное поле из него вытесняется

***Описание работы***

Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** В качестве примеси в германий ввели фосфор. Каким типом проводимости будет обладать полученный образец?

**ПРАКТИЧЕСКАЯ РАБОТА № 7. Физика атомного ядра и элементарных частиц**

***Цель занятия:*** изучить систему законов и принципов физики атомного ядра и элементарных частиц.

***Основные понятия***

**Адроны –** элементарные частицы, участвующие наряду с электромагнитным и слабым в сильном взаимодействии

**Зарядовое число ядра –** число протонов в ядре

**Изотопические мультиплеты –** подобные группы «похожих» элементарных частиц, одинаковым образом участвующих в сильном взаимодействии, имеющие близкие массы и отличающиеся зарядами

**Изотопы –** ядра с одинаковыми зарядовыми числами, но разными массовыми числами

**Критическая масса –** минимальная масса делящегося вещества, находящегося в системе критических размеров, необходимая для осуществления цепной реакции

**Массовое число ядра –** общее число протонов и нейтронов в атомном ядре

**Период полураспада –** время, за которое исходное число радиоактивных ядер в среднем уменьшается вдвое

**Поглощенная доза излучения –** физическая величина, равная отношению энергии излучения к массе облучаемого вещества

**Радиоактивное семейство –** совокупность элементов, образующих цепочку радиоактивных превращений, заканчивающихся стабильным элементом

**Радиоактивный распад –** естественное радиоактивное превращение ядер, происходящее самопроизвольно

**Реакция деления ядра –** ядерная реакция, в которой тяжелое ядро под действием частиц (например, нейтронов) делится на несколько более легких ядер (осколков), чаще всего на два ядра, близких по массе

**Среднее время жизни элементарной частицы –** мера стабильности частицы, выраженная в секундах

**Термоядерные реакции –** реакции синтеза легких атомных ядер в более тяжелые, происходящие при сверхвысоких температурах (примерно $10^7$ К и выше)

**Удельная энергия связи –** энергия связи ядра, отнесенная к одному нуклону

**Фундаментальные частицы –** бесструктурные элементарные частицы, которые до настоящего времени не удалось описать как составные

**Характерное ядерное время –** время, необходимое для пролета частицей расстояния порядка величины, равной диаметру ядра

**Цепная реакция деления –** ядерная реакция, в которой частицы, вызывающие реакцию, образуются как продукты этой реакции

**Электрический заряд элементарной частицы –** физическая величина, характеризующая способность частицы участвовать в электромагнитном взаимодействии, выраженная в единицах элементарного заряда $e \approx 1{,}6 \cdot 10^{-19}$ Кл

**Элементарная частица –** микрообъект, который невозможно расщепить на составные части

**Энергия процесса –** характеристика любого превращения частиц, определяется как разность кинетических энергий конечных и начальных частиц в процессах взаимопревращения частиц

**Энергия связи ядра –** энергия, которую необходимо затратить, чтобы расщепить ядро на отдельные нуклоны

**Эффективное сечение –** физическая величина, характеризующая вероятность того, что при падении пучка частиц на вещество произойдет реакция; имеет размерность площади

**Эффект Мёссбауэра –** явление упругого испускания (поглощения) $\gamma$-квантов атомными ядрами, связанными в твердом теле, не сопровождающееся изменением внутренней энергии тела

**Ядерные реакции –** превращения атомных ядер при взаимодействии с элементарными частицами (в том числе и с $\gamma$ -квантами) или друг с другом

***Описание работы***

Выполнить задания.

Задания выполняются каждым обучающимся в отдельности.

***Задание 1*** В результате последовательной серии радиоактивных распадов $^{238}_{92}\mathrm{U}$ превращается в изотоп свинца $^{206}_{82}\mathrm{Pb}$. Определите, сколько α- и β-распадов испытывает исходный изотоп в данном радиоактивном семействе при его превращении в соответствующий изотоп свинца.

***Задание 2*** Натрий $^{23}_{11}\mathrm{Na}$, облучаемый дейтронами, превращается в радиоактивный изотоп натрия $^{24}_{11}\mathrm{Na}$ с периодом полураспада $T$ = 15,5 ч. Какая доля первоначального количества радиоактивного натрия останется через сутки, если прекратить облучение дейтронами?

***Задание 3*** При бомбардировке изотопа азота $^{14}_{7}\mathrm{N}$ нейтронами получается изотоп углерода $^{14}_{6}\mathrm{C}$, который оказывается β-радиоактивным. Запишите уравнения протекающих при этом ядерных реакций.

***Задание 4*** Вычислите дефект массы и энергию связи ядра кислорода. Известно, что $m_p$ = 1,00728 а.е.м., $m_n$ = 1,00866 а.е.м., $M_я$ = 16,99913 а.е.м.

***Задание 5*** Какая энергия выделяется при ядерной реакции $^{7}_{3}\mathrm{Li} + {}^{2}_{1}\mathrm{H} \rightarrow {}^{8}_{4}\mathrm{Be} + {}^{1}_{0}n$? Массы ядер соответственно равны: $^{2}_{1}\mathrm{H}$ – 2,01355 а.е.м.; $^{7}_{3}\mathrm{Li}$ – 7,01436 а.е.м.; $^{8}_{4}\mathrm{Be}$ – 8,00311 а.е.м.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**
**ПО ДИСЦИПЛИНЕ «ФИЗИКА»**
**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ**
**(РЕШЕНИЕ ЗАДАЧ)**

Ответственный за выпуск Е.Д. Кожевникова
Корректор М.В. Богданова
Оператор компьютерной верстки А.П. Митряхина

**4187.07.01;МУАТ.01;2**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО УЧЕБНОЙ ДИСЦИПЛИНЕ «МАТЕМАТИЧЕСКИЙ АНАЛИЗ»**
**ПО РАЗДЕЛУ «ЭЛЕМЕНТЫ ФУНКЦИОНАЛЬНОГО АНАЛИЗА»**

КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ
(РЕШЕНИЕ ЗАДАЧ)

**МОСКВА 2018**

Разработано Б.П. Осиленкером, д. ф.-м. н., проф.

Рекомендовано Учебно-методическим советом в
качестве методического пособия для обучающихся

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО УЧЕБНОЙ ДИСЦИПЛИНЕ «МАТЕМАТИЧЕСКИЙ АНАЛИЗ»**
**ПО РАЗДЕЛУ «ЭЛЕМЕНТЫ ФУНКЦИОНАЛЬНОГО АНАЛИЗА»**

КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ
(РЕШЕНИЕ ЗАДАЧ)

Методические указания (МУ) подготовлены для педагогических работников по проведению коллективного тренинга по дисциплине «Математический анализ». МУ являются неотъемлемой частью дидактического обеспечения проведения практических занятий и нацелены на формирование общекультурных, общепрофессиональных, профессиональных или компетенций у обучающихся.

**О Г Л А В Л Е Н И Е**

## I ВВЕДЕНИЕ

**Модуль:** «Элементы функционального анализа».

**Цель практических занятий** - ввести обучающегося в круг математических знаний, составляющих основы профессиональной культуры любого современного специалиста.

- **Задачи практических занятий:**
- воспитание достаточно высокой математической культуры;
- привитие навыков современных видов математического мышления;
- привитие навыков использования математических методов и основ математического моделирования в практической деятельности;
- обучение методам анализа опытных данных, результаты которых случайны;
- обучение статистически обрабатывать и систематизировать имеющуюся информацию;
- обучение сводить задачи принятия решений к математическим моделям;
- обучение математическим методам, использующимся при моделировании;
- формирование у обучающихся умения получать количественное обоснование принимаемых решений;
- развитие навыков самостоятельной работы;
- повышение мотивации к процессу изучения учебной дисциплины.
- **В результате изучения дисциплины обучающийся должен**

**знать**

основные понятия:

- метрического пространства и линейного нормированного пространства;
- Гильбертова пространства;
- ряды Фурье по ортогональным системам и нахождение элемента наилучшего приближения;
- линейные операторы и линейные функционалы в линейных нормированных пространствах;
- приложения к интегральным уравнениям.

## II УЧЕБНО-МЕТОДИЧЕСКОЕ, МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ

### а) Литература

**Основная учебная**

1 **Арефьев, В.Н.,** Логинов, Э.А. Дифференциальное исчисление функции одной переменной. [Электронный ресурс]: рабочий учебник/ Арефьев, В.Н., Логинов, Э.А. - 2012. - http://lib.muh.ru.

2 **Арефьев, В.Н.,** Логинов, Э.А. Интегральное исчисление функций одной переменной. [Электронный ресурс]: рабочий учебник/ Арефьев, В.Н., Логинов, Э.А. - 2012. - http://lib.muh.ru.

3 **Хавинсон, С.Я.,** Глаголева, Р.Я., Россовский, Л.Е., Кошелева, Е.Л., Логинов, Э.А. Дифференциальное и интегральное исчисление функций нескольких переменных.[Электронный ресурс]: рабочий учебник/ Хавинсон, С.Я., Глаголева, Р.Я., Россовский, Л.Е., Кошелева, Е.Л., Логинов, Э.А. - 2012. - http://lib.muh.ru

4 **Рубинштейн, А.И.**Обыкновенные дифференциальные уравнения. [Электронный ресурс]: рабочий учебник/ Рубинштейн, А.И. - 2012. - http://lib.muh.ru.

**Дополнительная**

1 **Шмелькин, Д.А**. Элементы теории функций и функционального анализа. [Электронный ресурс]: рабочий учебник/ Шмелькин, Д.А. - 2010. - http://lib.muh.ru

2 **Веретенников, В.Н**. Сборник задач по математике [Электронный ресурс]: методический материал/ Веретенников В.Н.— Электрон. текстовые данные.— СПб.: Российский государственный гидрометеорологический университет, 2011.— 340 с.— http://www.iprbookshop.ru/17964.— ЭБС «IPRbooks»

3 **Веретенников, В.Н**. Высшая математика [Электронный ресурс]: учебное пособие/ Веретенников В.Н.— Электрон. текстовые данные.— СПб.: Российский государственный гидрометеорологический университет, 2013.— 254 с.— http://www.iprbookshop.ru/17901.— ЭБС «IPRbooks»

4 **Арефьев, В.Н.,** Логинов, Э.А. Введение в математический анализ. [Электронный ресурс]: рабочий учебник/ Арефьев, В.Н., Логинов, Э.А. - 2011. - http://lib.muh.ru.

5 **Гунько, Ю.А**. Математический анализ [Электронный ресурс]: учебное пособие/ Гунько Ю.А.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2008.— 151 c.— http://www.iprbookshop.ru/11335.— ЭБС «IPRbooks».

**б) Информационное обеспечение**

– Ресурсы информационно-телекоммуникационной сети Интернет:

– - http://www.edu.ru/

– - http://www.math.ru/.

– Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы.
- тренинговые и тестирующие программы.
- интеллектуальные роботизированные системы оценки качества выполненных работ.

– Информационные и роботизированные системы, программные комплексы, программное обеспечение для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ПО «Комбат»;
- ПО «ЛиК»;
- ПК «КОП»;
- ИР «Каскад».

**в) Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

## III РЕШЕНИЯ ТИПОВЫХ ЗАДАЧ

### Тип 1. Определение метрического и нормированного пространств. Открытые и замкнутые множества

Говорят, что на множестве $X$ задана *метрика*, если имеется функция, сопоставляющая каждой паре элементов $x, y \in X$ неотрицательное вещественное число $\rho(x, y)$ и обладающая следующими свойствами:

а) $\rho(x, y) = 0$ тогда и только тогда, когда $x = y$ (аксиома тождества);

б) $\rho(y, x) = \rho(x, y)$ (аксиома симметрии);

в) для любых элементов $x, y, z \in X$ выполняется $\rho(x, y) + \rho(y, z) \geq \rho(x, z)$ (аксиома треугольника).

Множество $X$ вместе с введенной метрикой называется *метрическим пространством* и обозначается $(X, \rho)$ (или просто $X$).

Открытым шаром в метрическом пространстве $X$ с центром в точке $x \in X$ и радиусом $\varepsilon$ называется подмножество $U(x, \xi) = \{y \in X : \rho(x, y) < \xi\}$. Будем его также называть $\varepsilon$ - окрестностью точки $X$.

*Нормой* на линейном пространстве $X$ назовем функцию, сопоставляющую каждому элементу $x \in X$ число $\|x\|$ со следующими свойствами:

а) $\|x\| \geq 0$, причем $\|x\| = 0$ тогда и только тогда, когда $x = 0$;

б) для любого числа $\alpha$ : $\|\alpha x\| = |\alpha| \|x\|$;

в) неравенство треугольника: $\|x + y\| \leq \|x\| + \|y\|$.

Линейное пространство, снабженное нормой, называется линейным *нормированным пространством*.

Равенство $\rho(x, y) = \|x - y\|$ задает в $X$-метрику.

*Примеры.*

1. Пространство $l_p (p \geq 1)$: $\|\{x_i\}\| = \left( \sum_{i=1}^{\infty} |x_i|^p \right)^{\frac{1}{p}}$.

2. Пространство $C[a, b]$: $\|x\| = \max_{a \leq t \leq b} |x(t)|$.

3. Пространство $L_p[a, b]$: $\|x\| = \left( \int_a^b |x(t)|^p \, dt \right)^{\frac{1}{p}}$ $(1 \leq p < \infty)$.

Подмножество $Y$ метрического пространства $X$ называется открытым, если для любого $y \in Y$ существует открытый шар в $X$ с центром $y$, целиком содержащийся в $Y$.

Подмножество $B$ метрического пространства $X$ называется замкнутым, если оно содержит все свои предельные точки.

*Теорема*. Подмножество $B$ метрического пространства $X$ является замкнутым тогда и только тогда, когда его дополнение $X \setminus B$ является открытым множеством.

**Задача 1.** Найти норму элемента $x(t) = t^3 + 3t^2 + 2t$ в $C[0,3]$.

**Решение.**

$\|x\|_c = \max\limits_{0\le t\le 3}|x(t)|$;

$x(t) = t^3 + 3t^2 + 24$;

$x'(t) = 3t^2 + 6t = 3t(t+2) \rangle 0$;

$\|x\|_c = x(3) = 78$.

Ответ: $\|x\|_c = 78$.

**Задача 2.** Найти норму элемента $x(t) = e^{3t}$ в пространстве $L_2[\ln 2, \ln 3]$.

**Задача 3.** Найти расстояние между элементами $x_1 = t^2$ и $x_2 = t^3$: а) в пространстве $C[0,1]$, б) в пространстве $L_2[0,1]$.

**Задача 4.** Найти в пространстве $l_2$ расстояние между элементами $x = \left\{\frac{1}{2^n}\right\}(n = 1,2,...)$ и $y = \left\{\frac{1}{3^n}\right\}(n = 1,2,...)$.

**Задача 5.** Доказать, что множество $M$ в евклидовом пространстве $\Re^2$, задаваемое системой

$$\begin{cases} x + y \rangle 5 \\ x^2 + y^2 \langle 100 \end{cases},$$

**Тип 2. Гильбертово пространство**

*Скалярным произведением* на вещественном линейном пространстве $X$ назовем функцию, сопоставляющую каждой паре векторов $x, y \in X$ вещественное число $(x, y)$, со следующими свойствами:

а) $(x,x) \ge 0$, причем $(x,x) = 0$ тогда и только тогда, когда $x=0$;

в) $(y,x) = (x,y)$;

с) для любых $\alpha, \beta \in \Re$ выполняется $(\alpha x + \beta y, z) = \alpha(x,z) + \beta(y,z)$.

*Евклидовым пространством* назовем вещественное линейное пространство $X$, снабженное скалярным произведением.

Угол между векторами евклидова пространства вычисляют по формуле

$$\cos\left(\hat{xy}\right) = \frac{(x,y)}{\sqrt{(x,x)}\sqrt{(y,y)}}.$$

Норма на евклидовом пространстве задается формулой $\|x\| = \sqrt{(x,x)}$.

Евклидово линейное пространство, полное относительно нормы, согласованной со скалярным произведением, называется гильбертовым.

*Примеры гильбертовых пространств.*

1. Каждое конечномерное евклидово пространство является гильбертовым.

Например, в $\Re^2$ $(x,y)=\sum_{n=1}^{n}\xi_k\eta_k$.

2. Пространство $L_2[a,b]$, в котором $(f,g)=\int_a^b f(t)g(t)dt$ и $\|f\|_2=\left\{\int_a^b f^2(t)dt\right\}^{\frac{1}{2}}$.

3. Пространство $l_2$, в котором $(x,y)=\sum_{k=1}^{\infty}\xi_k\eta_k$ и $\|x\|_2=\left(\sum_{k=1}^{n}|\xi_k|^2\right)^{\frac{1}{2}}$.

Векторы *x* и *y* евклидова пространства *X* называются ортогональными, если $(x,y)=0$.

Система ненулевых векторов $x_1,x_2,...$ называется ортогональной, если любые два элемента системы ортогональны: $(x_i,x_j)=0(i\neq j)$.

Система ортогональных векторов называется ортонормированной, если норма каждого элемента равна единице.

Разделив каждый элемент ортогональной системы на его норму, получим ортонормированную систему.

Пусть $u_1,u_2,...,u_m$ – линейно независимые векторы гильбертова пространства *X*.

Процесс ортогонализации Грама-Шмидта:

$v_1=u_1$;

$v_2=u_2-\frac{(u_2,v_1)}{(v_1,v_1)}v_1$;

$v_3=u_3-\frac{(u_3,v_1)}{(v_1,v_1)}v_1-\frac{(u_3,v_2)}{(v_2,v_2)}v_2$;

- - - - - - - - - - - - - - - - - - - - - - -

$v_i=u_i-\sum_{k=1}^{i-1}\frac{(u_i,v_k)}{(v_k,v_k)}v_k\ (i=1,2,\ldots,m)$.

Результат процесса: векторы $v_1,v_2,\ldots,v_m$ образуют ортогональную систему.

**Задача 1.** Вычислить скалярное произведение элементов *f(x)=x* и $g(x)=e^x$ в пространстве $L_2[0,1]$.

**Решение.**

$(f,g)=\int_0^1 xe^x dx=xe^x\Big|_0^1-\int_0^1 e^x dx=1-e^x\Big|_0^1=1$.

Ответ: $(f,g)=1$.

**Задача 2.** Найти косинус угла между элементами $f(x)=x$ и $g(x)=e^x$ в пространстве $L_2[0,1]$.

**Задача 3.** Применяя алгоритм Грама-Шмидта, провести ортогонализацию системы векторов $u=\{2,-1,0\}$ и $v=\{1,3,-1\}$.

**Задача 4.** Применяя алгоритм Грама-Шмидта, провести ортогонализацию системы $u_1=\{1,1,0\}$, $u_2=\{3,-1,1\}$, $u_3=\{4,0,-1\}$ и пронормировать полученные векторы.

**Задача 5.** Выяснить, будут ли функции $\cos 2x$ и $\cos 5x$ ортогональны

а) в пространстве $L_2[-\pi,\pi]$;

в) в пространстве $L_2\left[-\frac{\pi}{4},\frac{\pi}{4}\right]$.

**Тип 3. Ряды Фурье по ортогональным системам. Нахождение элемента наилучшего приближения**

Пусть $\varphi_1,\varphi_2,...,\varphi_{k,...}$ – ортогональная система векторов гильбертова пространства $X$ и $x\in X$ – некоторый элемент.

Рядом Фурье элемента $x$ по ортогональной системе $\{\varphi_n\}(n=1,2,...)$ называется ряд

$$x\sim\sum_{k=1}^{\infty}c_k\varphi_k,\quad где\quad c_k=\frac{(x,\varphi_k)}{(\varphi_k,\varphi_k)}\quad (k=1,2,...)$$

или

$$x\sim\sum_{k=1}^{\infty}\hat{c}_k\hat{\varphi}_k,\quad где\quad \hat{c}_k=\left(x,\hat{\varphi}_k\right)\quad (k=1,2,...),$$

где $\hat{\varphi}_k=\frac{\varphi_k}{\|\varphi_k\|}$ $(k=1,2,...)$ – соответствующая ортонормированная система.

Обозначим через $S_n=\sum_{k=1}^{n}\hat{c}_k\hat{\varphi}_k$ $\left(S_n=\sum_{k=1}^{n}\frac{(x,\varphi_k)}{(\varphi_k,\varphi_k)}\varphi_k\right)$ – частичные суммы ряда Фурье элемента $x\in X$.

Если $L\subseteq X$ – пространство $X$, то элементом наилучшего приближения элемента $x\in X$ в подпространстве $L$ называется такой элемент $y\in L$, что $\|x-y\|=\rho(x,L)=\inf_{z\in L}\|x-z\|$.

Элемент наилучшего приближения – это основание перпендикуляра, опущенного из элемента $x$ на $L$.

В гильбертовом пространстве $X$ элементом наилучшего приближения для $x\in X$ в подпрост-ранстве $H_n$, порожденном $\varphi_1,\varphi_2,\varphi_3,...,\varphi_n$, является частичная сумма ряда Фурье элемента $x$:

$$\rho(x,L)=\|x-S_n\|,$$

где $S_n=\sum_{k=1}^{n}\frac{(x,\varphi_k)}{(\varphi_k,\varphi_k)}\varphi_k$.

1. В пространстве $L_2[a,b]$ задана ортогональная система функций $\varphi_1(x),\varphi_2(x),...,\varphi_n(x),....$

Тогда $\min_{\alpha_1,\alpha_2,...\alpha_n}\int_a^b\left[f(x)-\sum_{k=1}^{n}\alpha_k\varphi_k(x)\right]^2dx=\int_a^b\left[f(x)-\sum_{k=0}^{n}\frac{c_k}{(\varphi_k,\varphi_k)}\varphi_k(x)\right]^2dx$,

т.е. элемент наилучшего приближения:

$$\sum_{k=1}^{n}\frac{c_k}{(\varphi_k,\varphi_k)}\varphi_k(x),\quad c_k=\int_a^b f(x)\varphi_k(x)dx,\ (\varphi_k,\varphi_k)=\int_a^b\varphi_k^2(x)dx\ \ (k=1,2,...).$$

В частности, элемент наилучшего приближения (в смысле гильбертова пространства $L_2[-1,1]$) функции $f(x)$ в пространстве многочленов степени $\le n$:

$$\sum_{k=0}^{n}\frac{(2k+1)a_k}{2}P_k(t),\quad a_k=\int_{-1}^{1}f(t)P_k(t)dt\qquad (k=1,2,...n),$$

где $P_k(t)\quad (k=1,2,...)$ – многочлены Лежандра;

$$P_0(t)\equiv 1,\qquad P_1(t)=t,\quad P_2(t)=\frac{1}{2}\left(3t^2-1\right),....$$

2. Если $f$ – $2\pi$ - периодическая функция из $L_2[-\pi,\pi]$, то элемент наилучшего приближения (в смысле гильбертова пространства $L_2[-\pi,\pi]$) тригонометрическими многочленами степени $\le n$:

$$\frac{1}{2}a_0+\sum_{k=1}^{n}(a_k\cos kt+b_k\sin kt),$$

где $a_k=\frac{1}{\pi}\int_{-\pi}^{\pi}f(t)\cos ktdt,\qquad (k=0,1,2,...),\ b_k=\frac{1}{\pi}\int_{-\pi}^{\pi}f(t)\sin ktdt,\qquad (k=1,2,...).$

**Задача 1.** Найти коэффициент Фурье элемента $f(t)=t^2(-\pi\le t\le\pi)$ по ортогональной системе $\{1,\cos nt,\sin nt\}(n=4,2,...)$ при $\cos 2t$.

**Решение.**

$$a_2=\frac{2}{\pi}\int_0^{\pi}t^2\cdot cos\,2tdt=\frac{2}{\pi}\left\{\frac{1}{2}t^2\cdot sin\,2t\ \Big|_0^{\pi}-\int_0^{\pi}t\cdot sin\,2tdt\right\}=\frac{2}{\pi}\left\{\frac{1}{2}t\,cos\,2t\ \Big|_0^{\pi}-\frac{1}{4}sin\,2t\ \Big|_0^{\pi}\right\}=1.$$

Ответ: $a_2=1$.

**Задача 2.** Найти элемент наилучшего приближения (в смысле гильбертова пространства $L_{2\pi}^2[-\pi,\pi]$) для функции $f(t)=-t(-\pi\le t\le\pi)$ в пространстве тригонометрических многочленов степени меньше либо равной 2.

**Задача 3.** Найти элемент наилучшего линейного приближения функции $f(t)=3t^2$ (в смысле гильбертова пространства $L_2[-1,1]$) в пространстве многочленов.

**Задача 4.** Найти элемент наилучшего линейного приближения (в смысле гильбертова пространства $L_2[-1,1]$) функции $f(t)=e^t$ в пространстве многочленов.

**Тип 4. Линейные операторы и функционалы. Операторы сжатия**

1. Пусть $X$ и $Y$ – линейные нормированные пространства.

Оператор $A$: $X\to Y$ называется линейным, если для любых $x_1,x_2$ из области определения оператора $A$ и любого числа $\alpha$ выполняются аксиомы:

1) $A(x_1+x_2)=Ax_1+Ax_2$;

2) $A(\alpha x_1)=\alpha A(x_1)$.

Оператор $A$: $X \to Y$ называется ограниченным, если он каждое ограниченное подмножество отображает из $X$ в ограниченное подмножество из $Y$.

Пусть $A$: $X \to Y$ является линейным ограниченным оператором.

Нормой оператора $A$ называется число $\|A\| = \underset{x\in X, x\neq 0}{Sup} \dfrac{\|A(x)\|}{\|x\|} = \underset{x\in X, \|x\|\leq 1}{Sup} \|A(x)\|.$

Частным случаем линейного оператора является понятие линейного функционала.

Линейным вещественным функционалом называется отображение $f : X \to \Re^1$, при этом

1) $f(x_1 + x_2) = f(x_1) + f(x_2), \quad \forall x_1, x_2 \in X$;

2) $f(\alpha x_1) = \alpha f(x_1), \quad \forall x_1 \in X, \forall \alpha \in \Re^1$.

Норма линейного ограниченного функционала определяется следующим образом:

$$\|f\| = \underset{x\in X, x\neq 0}{Sup} \frac{|f(x)|}{\|x\|} = \underset{x\in X, \|x\|\leq 1}{Sup} |f(x)|.$$

2. Пусть задано отображение метрического пространства $X$ в себя.

Отображение $\varphi : X \to Y$ называется сжатым (сжатием), если существует такое число $q\langle 1$, называемое коэффициентом сжатия, что для любых $x, y \in X$ выполняется $\rho(\varphi(x), \varphi(y)) \leq q\rho(x, y)$.

Линейным оператор

$A : X \to X$ является сжатым отображением тогда и только тогда, когда он ограничен и его норма $\|A\| = q\langle 1$.

*Пример.*

Пусть числовая функция $f$ отображает отрезок $[a, b] \to \Re^1$.

Если $f(x)$ дифференцируемая и для всех $x \in [a, b]$ выполнено $|f'(x)| \leq q\langle 1$, то отображение $f$ является сжатым (сжатием) с коэффициентом сжатия $q$.

**Задача 1.** Доказать, что оператор $A : C[0,1] \to C[0,1]$,

$$Ax(t) = \int_0^t x(s)ds$$

является линейным ограниченным и найти его норму.

**Решение.**

1: а) $A[x_1 + x_2](t) = \int_0^t [x_1(s) + x_2(s)]ds = \int_0^t x_1(s)ds + \int_0^t x_2(s)ds = Ax_1(t) + Ax_2(t)$;

в) $A(\lambda x)a) = \lambda Ax(t)$ – аналогично.

2: $\|Ax\|_c = \max_{0\leq t\leq 1} \left| \int_0^t x(s)ds \right| \leq \max_{0\leq t\leq 1} |x(s)| \cdot \max_{0\leq t\leq 1} t \leq 1 \cdot \|x\|_c$.

Значит, $\|A\| \leq 1$;

Рассмотрим функцию $x_0(t) \equiv 1(t \in [0,1])$;

$\|x_0\|_c = 1$.

Имеем $\|Ax_0\|_c = \max\limits_{0\le t\le 1}\left|\int\limits_0^t x_0(s)ds\right| = 1\cdot\max\limits_{0\le t\le 1} t = 1\cdot\|x_0\|$.

Следовательно, $\|A\|\ge 1$.

Ответ: $A$ линейный ограниченный оператор с нормой $\|A\| = 1$.

**Задача 2.** Доказать, что оператор $Ax(t) = \int\limits_0^{\pi} sin\,t\cdot cos\,s\cdot x(s)\ ds$ является линейным и действует из $L_2[0,\pi]$ в $L_2[0,\pi]$. Найти оценку его нормы.

**Задача 3.** Доказать, что функционал $f$, заданный на $l_2$ $x=\{\xi_n\}(n=1,2,...)$, $x\in l_2 \to f(x) = \sum\limits_{n=1}^{\infty}\frac{\xi_n}{2^n}$ является линейным ограниченным и действует из $l_2$ в $l_2$. Найти его норму.

**Задача 4.** Проверить, что функция $f(x) = x^2$ отображает отрезок $\left[0,\frac{1}{3}\right]$ в себя, и это отображение является сжимающим.

**Задача 5.** Доказать, что функция $\varphi(t) = 1+\frac{1}{2}arctgt$ отображает $\Re^1$ в себя и является сжатием.

**Тип 5. Собственные значения и собственные векторы линейного оператора**

Собственным вектором линейного оператора $A: X\to X$ называется такой ненулевой вектор, что $Ax = \alpha x$ для некоторого комплексного числа $\alpha$.

При этом число $\alpha$ называется собственным значением линейного оператора $A$, соответствующим собственному вектору $x$.

Характеристическим числом линейного оператора $A$ назовем такое комплексное число $\lambda$, что уравнение $x-\lambda Ax = 0$ имеет ненулевое решение $x\in X$.

Таким образом, характеристические числа линейного оператора суть обратные числа к собственным значениям, отличным от нуля.

Назовем регулярным числом линейного оператора $A$ такое комплексное число $\beta$, что уравнение $x-\beta Ax = y$ имеет единственное решение для любого $y$.

Спектром линейного оператора $A$ называется совокупность всех комплексных чисел, за исключением регулярных.

Линейный оператор $A$, отображающий гильбертово пространство $x$ в себя, называется самосопряженным, если $A$ определен на всем $x$ и для любых $x, y\in x$ выполнено $(Ax, y) = (x, Ay)$.

1. Пусть $x = C^n$ (или $\Re^n$) со стандартным скалярным произведением и оператор $A$ задается матрицей с комплексными (вещественными) элементами. Тогда оператор $A$ самосопряжен, если его матрица является эрмитовой (симметричной).

2. В случае интегрального оператора с вещественным ядром $Ax(t)=\int_a^b K(t,s)x(s)ds$ симметричность оператора (действующего, например, из $L_2[a,b]$ в себя) означает симметричность ядра: $K(s,t)=K(t,s)$.

Собственные значения линейного самосопряженного оператора вещественны.

Спектр самосопряженного компактного (вполне непрерывного) оператора, действующего в гильбертовом пространстве $\boldsymbol{X}$, состоит из конечного или счетного множеств собственных значений, единственной предельной точкой которых может служить точка $\alpha=0$.

**Задача 1.** Найти спектр, характеристические и регулярные значения оператора *A*, заданного в евклидовом пространстве $\Re^2$ матрицей $\begin{pmatrix}5 & 6\\ 6 & 0\end{pmatrix}$. Найти собственные векторы.

**Решение.**

Найдем собственные значения $\begin{vmatrix}5-\lambda & 6\\ 6 & -\lambda\end{vmatrix}=0,\quad \lambda_1=9,\quad \lambda_2=-4$.

Собственные векторы:

1) $\lambda_1=9\to 2\xi_1=3\xi_2\to x_1=\{3,2\}$;

$x_1\perp x_2$;

2) $\lambda_2=-4\to 3\xi_1=-2\xi_2\to x_2=\{2,-3\}$.

Характеристические числа: $\frac{1}{9},-\frac{1}{4}$.

Спектр: $\sigma(A)=\left\{\frac{1}{9},-\frac{1}{4}\right\}$.

Регулярные числа: $\Re^1\setminus\sigma(A)$, т.е. $\left(-\infty,-\frac{1}{4},\right)\cup\left(-\frac{1}{4},\frac{1}{9}\right)\cup\left(\frac{1}{9},\infty\right)$.

**Задача 2.** Найти собственные значения, характеристические числа и собственные функции краевой задачи: $y''=-\lambda y,\ y(0)=y(1)=0$.

**Задача 3.** В вещественном пространстве $C[0,l]$ для интегрального оператора Фредгольма $Ax(t)=\int_0^l K(t,s)x(s)ds$ с ядром $K(t,s)=\begin{cases}\frac{t(l-s)}{l} & при\quad t\le s\\ \frac{s(l-t)}{l} & при\quad t\ge s\end{cases}$ $0\le t,s\le l$ найти характеристические числа и ортонормированные собственные функции.

**Задача 4.** Пусть *A* – оператор дифференцирования, действующий из линейного многообразия непрерывно дифференцированных функций в $C[a,b]$. Найти его собственные числа и собственные функции.

**Задача 5.** Пусть *x* – собственный вектор оператора *A*, отвечающий собственному значению $\lambda$. Тогда *x* – собственный вектор оператора $A^2$ с собственным значением $\lambda^2$. Если $\lambda\neq 0$, то *x* – собственный вектор оператора $A^{-1}$ с собственным значением $\frac{1}{\lambda}$.

**Тип 6. Линейные интегральные уравнения**

1. Типы линейных интегральных уравнений:

1) Фредгольма 1 рода $\int_a^b K(t,s)x(s)ds = f(t)$;

2) Фредгольма 2 рода $x(t) - \int_a^b K(t,s)x(s)ds = f(t)$;

3) Вольтерра 1 рода $\int_a^t K(t,s)x(s)ds = f(t) \quad (a \le t \le b)$;

4) Вольтерра 2 рода $x(t) - \int_a^t K(t,s)x(s)ds = f(t)$.

Здесь $x(t)$ – искомая функция; $K(t,s); f(t)$ – заданные функции; $K(t,s)$ –ядро; $f(t)$ –свободный член.

Решением интегрального уравнения называется всякая функция $x(t)$, $(a \le t \le b)$, подстановка которой в это уравнение обращает его в тождество.

2. Метод последовательных приближений.

А. Для интегрального уравнения Вольтерра второго рода:

$$x_n(t) = f(t) + \int_a^t K(t,s)x_{n-1}(s)ds, \ (n = 0,1,2\ldots).$$

Если ядро $K(t,s)$ и свободный член $f(t)$ непрерывны соответственно при $a \le t \le b, \ a \le s \le x$ и на отрезке $[a,b]$, то построенная таким образом последовательность приближений $x_n(t)$, $n = 0,1,\ldots$, при $n \to \infty$ сходится к единственному непрерывному решению интегрального уравнения.

Б. Обычно рассматривается не одно уравнение, а семейство уравнений

$$x(t) - \lambda\int_a^b K(t,s)x(s)ds = f(t),$$

зависящих от числового параметра $\lambda$.

Рассмотрим случай, когда $|\lambda| < \frac{1}{B}, \qquad B = \left(\int_a^b\int_a^b |K(t,s)|^2 dtds\right)^{\frac{1}{2}} < \infty$.

Последовательные приближения строятся следующим образом:

$$x_n(t) = f(t) + \int_a^b K(t,s)x_{n-1}(s)ds.$$

При $n \to \infty$ эта последовательность сходится к единственному решению интегрального уравнения Фредгольма [непрерывному, если непрерывны $K(t,s)$ и $f(t)$].

3. Уравнения Фредгольма второго рода с вырожденным ядром.

Ядро $K(t,s)$ называется вырожденным, если $K(t,s)=\sum_{j=1}^{n} p_j(t)q_j(s)$.

Соответствующее интегральное уравнение $x(t)-\mu\int_a^b\left(\sum_{j=1}^{n} p_j(t)q_j(s)\right)x(s)dt=f(t)$ решают путем сведения к системе линейных алгебраических уравнений:

$$S_j=\mu\sum_{j=1}^{n}K_{ij}S_j+C_j,$$

где $S_j=\int_a^b q_j(s)x(s)ds, \quad (j=1,2,...,n);$

$$K_{ij}=\int_a^b p_i(s)q_j(s)ds, \quad (i,j=1,2,...,n);$$

$$C_j=\int_a^b f(s)q_j(s)ds, \quad (j=1,2,...,n).$$

Решение интегрального уравнения имеет вид $x(t)=\mu\sum_{i=1}^{n}S_iP_i(t)+f(t)$.

**Задача 1.** Является ли функция $x(t)=\frac{3}{2}t$ решением интегрального уравнения $x(t)-\int_0^1 tsx(s)ds=t$ ?

**Решение.**

$$\frac{3}{2}t-t\int_0^1\frac{3}{2}s\cdot sds=\frac{3}{2}t-t\cdot\frac{3}{2}\cdot\frac{1}{3}=t$$

Ответ: да.

**Задача 2.** Выяснить, применим ли метод последовательных приближений к интегральному уравнению Фредгольма второго рода: $x(t)=1+\int_0^1 ts^2x(s)ds$ и найти первых три приближения.

**Задача 3.** Найти решение интегрального уравнения Фредгольма второго рода:

$$x(t)-\frac{1}{\pi}\int_0^{2\pi}\cos t\sin s\cdot x(s)dt=\sin t\ .$$

**Задача 4.** Найти решение интегрального уравнения Фредгольма второго рода:

$$x(t)-\mu\int_{-1}^{1}(t+s)x(s)ds=\frac{1}{2}+\frac{3}{2}t\ .$$

**Задача 5.** Найти решение интегрального уравнения Фредгольма второго рода:

$$x(t)-\mu\int_0^{\pi}cos(t+S)x(S)dS=cos\,3t\ .$$

## IV ЗАКЛЮЧИТЕЛЬНАЯ ЧАСТЬ

В заключительной части коллективного алгоритмического тренинга преподаватель делает выводы, где отмечает положительные и отрицательные моменты в проведении практического занятия, а также дает краткие указания, советы по подготовке к следующему занятию.

В конце нужно оставить время для ответов на вопросы, возникшие у обучающихся.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО УЧЕБНОЙ ДИСЦИПЛИНЕ «МАТЕМАТИЧЕСКИЙ АНАЛИЗ»**
**ПО РАЗДЕЛУ «ЭЛЕМЕНТЫ ФУНКЦИОНАЛЬНОГО АНАЛИЗА»**

КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ
(РЕШЕНИЕ ЗАДАЧ)

Ответственный за выпуск Е.Д. Кожевникова
Корректор Т.М. Афонина
Оператор компьютерной верстки Е.Д. Кожевникова

**4184.01.01;МУ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**
**ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»**
**(для направления подготовки 09.03.01 «Информатика**
**и вычислительная техника»)**

**МОСКВА 2018**

Разработано Е.В. Корнеевой
Под редакцией Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических пособий
для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ» (для направления подготовки 09.03.01 «Информатика и вычислительная техника»)

Методические указания по проведению практических занятий по дисциплине «Программирование», направление 09.03.01 «Информатика и вычислительная техника», предназначены для закрепления и дополнения знаний, полученных на лекциях и практических занятиях.

**О Г Л А В Л Е Н И Е**

## 1 Введение

Методические материалы представляют собой комплекс лабораторных практикумов для аудиторной работы, а также указаний и разъяснений, позволяющих обучающемуся сформировать знания в области алгоритмизации и программирования, и выполнить задания, предусмотренные программой дисциплины.

Настоящие методические указания по выполнению лабораторных практикумов по курсу «Программирование» составлены на основе требований Федерального государственного образовательного стандарта высшего образования .

Основные задачи практических заданий направлены на то, чтобы познакомить обучающих с:

- методами программирования линейных, разветвляющихся и циклических алгоритмов;
- приемами программирования с использованием массивов и подпрограмм.

В практических работах оцениваются владение знаниями в области алгоритмизации и программирования.

## 2 ПРАКТИЧЕСКАЯ РАБОТА № 1. ПРОГРАММИРОВАНИЕ ЛИНЕЙНЫХ АЛГОРИТМОВ

***Тема:*** программирование линейных алгоритмов.

***Цель занятия:*** сформировать у обучаемых навыки программирования линейных алгоритмов.

***Основные понятия***

**Алгоритм** - точное предписание, определяющее вычислительный процесс, ведущий от варьируемых начальных данных к искомому результату.

**Алгоритм линейной структуры (следование)** - алгоритм, в котором все действия выполняются последовательно друг за другом.

**Язык программирования** - система обозначений для точного описания алгоритмов для ЭВМ.

**Программа** — текст, задающий множество процессов вычислений, в соответствии с которым исполнитель, понимающий программу, разворачивает какой-то один из них.

**Программирование** - процесс определения последовательности инструкций, которые должен выполнить компьютер для решения определенной задачи.

**Система программирования** - совокупность языка программирования и виртуальной машины, обеспечивающей выполнение реальной машиной программ, составленных на этом языке.

***Вводная часть***

Программы с линейной структурой составляются из операторов присваивания, ввода, вывода, обращения к процедурам. Оператор присваивания можно назвать основным в любом языке программирования.

*Оператор присваивания*: <переменная> := <выражение>

Оператор выполняется следующим образом. Вычисляется значение <выражения>, после чего <переменная> получает вычисленное значение. При этом тип выражения должен быть совместим с типом переменной.

Примеры оператора присваивания:

```
X:=(Y+Z)/(2+Z*10)-1/3;
LogPer:=(A>B) and (C<=D).
```

*Выражение* может включать в себя константы, переменные, знаки операций, функции, скобки. В результате вычисления выражения получается значение определенного типа.

Тип выражения определяется типом полученного значения.

*Арифметическое выражение* — выражение числового типа (целого или вещественного). Идентификатор целого типа: integer, вещественного типа: real.

*Арифметические операции* бывают унарными и бинарными. К унарным относится операция изменения знака. Ее формат: - <величина>.

В следующей таблице представлены бинарные арифметические операции Паскаля. A и B обозначают операнды, для типов величин использованы обозначения: I — целый, R — вещественный.

| Выражение | Тип операндов | Тип рез-та | Операция |
|---|---|---|---|
| A + B | R, R | R | Сложение |
| | I, I | I | |
| | I, R R, I | R | |
| A - B | R, R | R | Вычитание |
| | I, I | I | |
| | I, R R, I | R | |
| A * B | R, R | R | Умножение |
| | I, I | I | |
| | I, R R, I | R | |
| A / B | R, R | R | Вещественное деление |
| | I, I | R | |
| | I, R R, I | R | |
| A div B | I, I | I | Целое деление |
| A mod B | I, I | I | Остаток от целого деления |

*Стандартные математические функции Паскаля* представлены в следующей таблице:

| Обращение | Тип аргумента | Тип рез-та | Функция |
|---|---|---|---|
| abs(x) | I, R | I, R | Модуль x |
| arctan(x) | I, R | R | Арктангенс (x в радианах) |
| cos(x) | I, R | R | Косинус (x в радианах) |
| exp(x) | I, R | R | $e^x$ — экспонента x |
| frac(x) | I, R | R | Дробная часть x |
| int(x) | I, R | R | Целая часть x |
| ln(x) | I, R | R | Натуральный логарифм x |
| random | | R | Псевдослучайное число в интервале [0,1] |
| Обращение | Тип аргумента | Тип рез-та | Функция |
| random(x) | I | I | Псевдослучайное число в интервале [0,x] |
| round(x) | R | I | Округление x до ближайшего целого |
| sin(x) | I, R | R | Синус (x — в радианах) |
| sqr(x) | I, R | R | Квадрат x |
| sqrt(x) | I, R | R | Корень квадратный x |
| trunc(x) | R | I | Ближайшее целое, не превышающее x по модулю |

Старшинство операций (в порядке убывания приоритета):

=> вычисление функции;

=> унарный минус;

=> *, /, div, mod;

=> +, -

*Возведение положительного числа в вещественную степень* следует производить, используя следующее математическое тождество: $x^y = e^{y \ln x}$. На Паскале это записывается так:

exp (y*ln(x))

***Пример 1.*** Записать математические выражения в виде арифметических выражений на Паскале.

| Математическое выражение | Выражение на Паскале |
|---|---|
| $x^2 - 7x + 6$ | sqr(x)-7*x+6 |
| $\frac{\lvert x\rvert - \lvert y\rvert}{1 + \lvert xy\rvert}$ | (abs(x)-abs(y))/(1+abs(x*y)) |
| $ln\left\lvert\left(y - \sqrt{\lvert x\rvert}\right)\left(x - \frac{y}{z + \frac{x^2}{4}}\right)\right\rvert$ | ln(abs((y-sqrt(abs(x)))*(x-y/(z+sqr(x)/4)))) |

*Ввод данных с клавиатуры* производится путем обращения к стандартным процедурам:

read(<список ввода>) readln(<список ввода>)

Элементы списка ввода — идентификаторы переменных. Вводимые значения отражаются на экране. При выполнении оператора пользователь набирает на клавиатуре соответствующую последовательность значений, разделяя их пробелами.

*Вывод данных на экран* производится путем обращения к стандартным процедурам:

write(<список вывода>) writeln(<список вывода>)

Элементы списка вывода — константы, переменные, выражения, форматы вывода.

*Структура программы на Паскале:*

Program <Имя программы>;
Label <раздел описания меток>;
Const <раздел описания констант>;
Type <раздел описания типов>;
Var <раздел описания переменных>;
Procedure (Function) <раздел описания подпрограмму
Begin
<раздел операторов>
End.

Для любой программы обязательным является лишь раздел операторов. Все программные объекты (константы, переменные, типы и пр.) должны быть описаны в соответствующих разделах описаний.

***Пример 2.*** Скорость первого автомобиля $v_1$ км/ч, второго — $v_2$ км/ч, расстояние между ними 8 км. Какое расстояние будет между ними через $t$ ч, если автомобили движутся в разные стороны?

***Решение.*** Согласно условию задачи искомое расстояние $s_1 = s + (v_1 + v_2)t$ (если автомобили изначально двигались в противоположные стороны) или $s_2 = \lvert(v_1 + v_2)t - s\rvert$ (если автомобили первоначально двигались навстречу друг другу).

Программа организует ввод исходных данных, вычисление искомых величин по формулам и вывод их на экран. Все величины в программе - вещественного типа.

```
Program Car;
Var V1,V2,T,S,S1,S2 : Real;
Begin
```

```
  Write('Введите скорости автомобилей, расстояние между ними и время движения: ');
  ReadLn(V1,V2,S,T);
  S1:=S+(V1+V2)*T;
  S2:=Abs((V1+V2)*T-S);
  WriteLn('Расстояние будет равно ',S1:7:4, 'км или ',S2:7:4, ' км');
End.
```

***Пример 3.*** Выполнить вычисление по формуле $z = \left(\frac{t^2 - k^2}{m^4 - l^5}\right)^2 + \sqrt{\left|\frac{\sqrt{x} + y}{12 - |x|} + 4\right|}$.

***Решение***

```
Program Expression;
Var T,K,M,L,X,Y,Z : Real;
Begin
  Write('Введите значения переменных T,K,M,L,X,Y: ');
  ReadLn(T,K,M,L,X,Y);
  Z:=Sqr((T*T-K*K)/(Sqr(Sqr(M))-Exp(5*Ln(L))))+Sqrt(Abs((Sqrt(X)+Y)/(12-Abs(X))+4));
  WriteLn('Значение выражения: 'Z:12:6);
End.
```

*Логические выражения* в результате вычисления принимают логические значения true или false. Операндами логического выражения могут быть логические константы, переменные логического типа, отношения. Идентификатор логического типа в Паскале: boolean.

*Логические операции.* В Паскале имеются 4 логические операции: отрицание — NOT, логическое умножение — AND, логическое сложение — OR, исключающее «или» - XOR. Результаты логических операций для различных значений операндов приведены в таблице. Использованы обозначения: T — true, F — false.

| A | B | Not | A and B | A or B | A xor B |
|---|---|---|---|---|---|
| T | T | F | T | T | F |
| T | F | F | F | T | T |
| F | F | T | F | F | F |
| F | T | T | F | T | T |

Приоритеты логических операций:

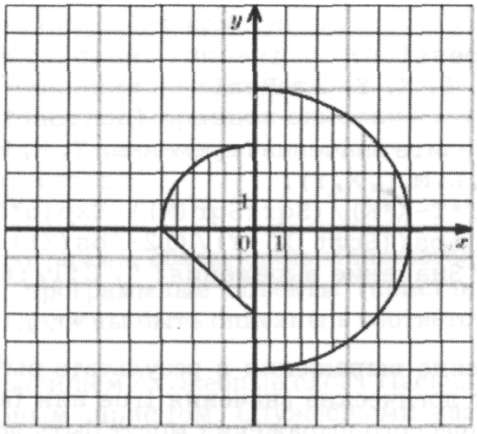
1) not;
2) and;
3) or;
4) xor.

Примеры логических выражений:

1) True;
2) False;
3) A>B;
4) (A=B) and (C<=0) .

Операции отношений (= , <>, >=, <=, <, >) имеют более низкий приоритет, чем логические операции, поэтому их следует заключать в скобки при использовании по отношению к ним логических операций.

***Пример 4.*** Составить программу, по которой выведется значение true, если точка с заданными координатами (x,y) лежит внутри заштрихованной области, и false - в противном случае.




***Решение.*** Рассматриваемая область состоит из двух частей, каждая из которых описывается системой неравенств.

1-я часть: $x < 0$; $x^2 + y^2 < 9$; $y > -x - 3$.

2-я часть: $x > 0$; $x^2 + y^2 < 25$.

Точка с координатами (x,y) лежит в заштрихованной области, если она принадлежит 1-й или 2-й части.

Программа вводит координаты точки, вычисляет логическое выражение, определяющее принадлежность точки области, и выводит полученную логическую величину на экран.

```
Program Point;
Var X,Y : real; L : boolean;
Begin
Write('X= '); ReadLn(X);
Write('Y= '); ReadLn(Y);
L:=(X<=0) and (Sqr(X)+Sqr(Y)<=9) and (Y>=-X-3) or (X>=0) and (Sqr(X)+Sqr(Y)<=25);
WriteLn('Точка лежит в заданной области? ',L);
End.
```

***Описание работы***

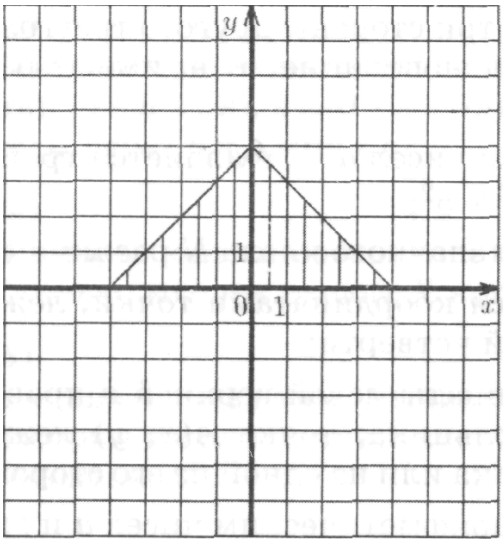
Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Составить программу, вычисляющую значение выражения (все переменные имеют действительный тип) $\frac{b+\sqrt{b^2+4ac}}{2a} - a^3c + b^{-2}$.

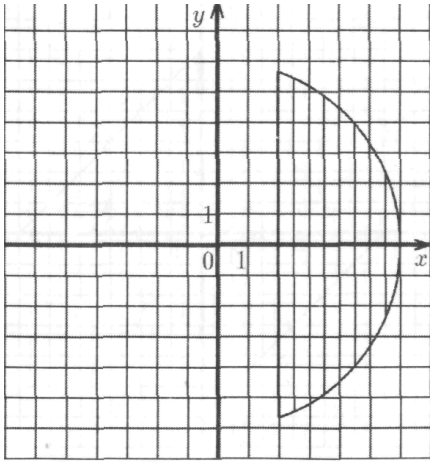
***Задание 2.*** Составить программу, вычисляющую значение выражения (все переменные имеют действительный тип) $\frac{\ln|\cos x|}{\ln(1+x^2)}$.

***Задание 3.*** Составить программу, вычисляющую периметр и площадь прямоугольного треугольника по заданным длинам двух катетов.

***Задание 4.*** Составить программу, которая печатает *true*, если точка с координатами *(x,y)* принадлежит заштрихованной области, и *false* в противном случае:




***Задание 5.*** Составить программу, которая печатает *true*, если точка с координатами *(x,y)* принадлежит заштрихованной области, и *false* в противном случае:




## 3 ПРАКТИЧЕСКАЯ РАБОТА № 2. ПРОГРАММИРОВАНИЕ РАЗВЕТВЛЯЮЩИХСЯ АЛГОРИТМОВ

***Тема:*** программирование разветвляющихся алгоритмов.

***Цель занятия:*** сформировать у обучаемых навыки программирования разветвляющихся алгоритмов.

***Основные понятия***

**Алгоритм** - точное предписание, определяющее вычислительный процесс, ведущий от варьируемых начальных данных к искомому результату.

**Алгоритм разветвляющейся структуры (ветвление)** – алгоритм, в котором предусмотрено разветвление указанной последовательности действий на два направления в зависимости от итога проверки заданного условия.

**Язык программирования** - система обозначений для точного описания алгоритмов для ЭВМ.

**Программа** — текст, задающий множество процессов вычислений, в соответствии с которым исполнитель, понимающий программу, разворачивает какой-то один из них.

**Программирование** - процесс определения последовательности инструкций, которые должен выполнить компьютер для решения определенной задачи.

**Система программирования** - совокупность языка программирования и виртуальной машины, обеспечивающей выполнение реальной машиной программ, составленных на этом языке.

***Вводная часть***

Для программирования разветвляющихся алгоритмов применяются условный оператор (оператор ветвления) и оператор выбора.

*Условный оператор* имеет следующий формат:

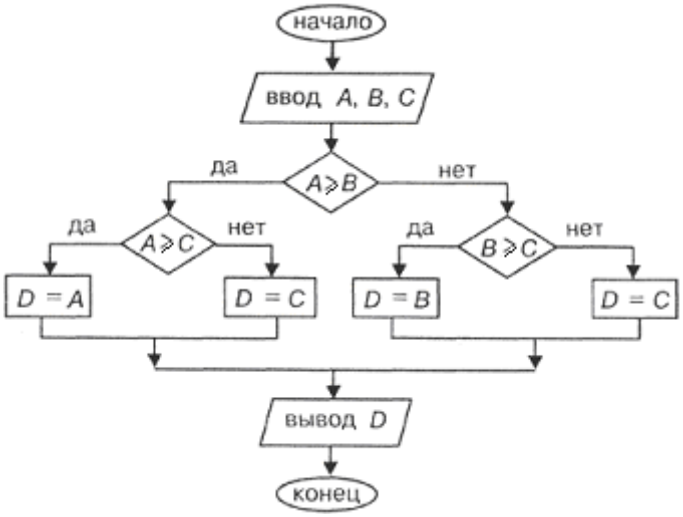
Операторы 1 и 2 могут быть простыми или составными. Если логическое выражение, выступающее в качестве условия ветвления, принимает значение False, то выполняется оператор 2, если True - оператор 1.

*Неполная форма условного оператора:*

if <логическое выражение> then <оператор>

***Пример 1.*** Из трех данных вещественных чисел A, B, C выбрать наибольшее.

***Алгоритм***




***Решение 1.*** Используем алгоритм с вложенными полными ветвлениями:

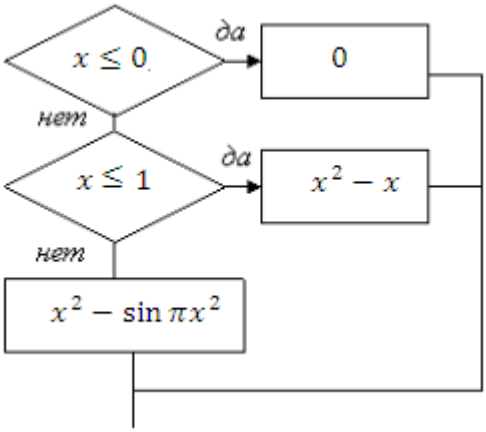
```
Program M1;
Var A,B,C,D : real;
Begin
  Write('A= '); ReadLn(A);
  Write('B= '); ReadLn(B);
  Write('C= '); ReadLn(C);
  If A>B
  Then If A>=C Then D:=A Else D:=C
  Else If B>=C Then D:=B Else D:=C;
  WriteLn('Максимальное значение=',D);
End.
```

***Решение 2.*** Используем алгоритм с последовательными неполными ветвлениями и сложными логическими выражениями.

```
Program M;
Var A,B,C,D : real;
Begin
  Write('A= '); ReadLn(A);
  Write('B= '); ReadLn(B);
  Write('C= '); ReadLn(C);
  If (A>=B) And (A>=C) Then D:=A;
  If (B>=A) And (B>=C) Then D:=B;
  If (C>=A) And (C>=B) Then D:=C;
  WriteLn('Максимальное значение=',D);
End.
```

***Пример 2.*** Дано действительное число *a*. Вычислить *f(a)*, если $f(x) = \begin{cases} 0, \text{если } x \le 0, \\ x^2 - x, \text{если } 0 < x \le 1, \\ x^2 - \sin \pi x^2, \text{если } x > 1. \end{cases}$

***Алгоритм***




***Решение.*** Алгоритм имеет вложенную ветвящуюся структуру.

```
Program F1;
Var X,F : real;
Begin
  Write('X= '); ReadLn(X);
  If (X<=0) Then F:=0
  Else If X<=1 Then F:=Sqr(X)-X
                Else F:=Sqr(X)-Sin(Pi*X*X);
WriteLN('X=',X,'  F=',F);
End.
```

*Оператор выбора* позволяет программировать ветвления по многим направлениям. Этот оператор организует переход на одну из нескольких ветвей в зависимости от значения заданного выражения (селектора выбора). Формат оператора выбора:

```
Case K Of
```

A1 : <оператор 1>;
A2 : <оператор 2>;
…
AN : <оператор N>
Else  <оператор N+1>
End;

Здесь К — выражение-селектор, которое может иметь только простой порядковый тип (целый, символьный, логический). A1,..., AN — константы того же типа, что и селектор, выполняющие роль меток ветвей. Исполнение оператора начинается с вычисления выражения К, полученное значение сравнивается с константами (метками) и выполняется соответствующий оператор. Если ни одна из меток не совпала со значением К, то выполняется оператор после слова Else.

Возможно использование неполного оператора выбора без ветви Else. Метки ветвей также могут быть заданы списком или интервалом.

***Пример 3.*** В старояпонском календаре был принят двенадцатилетний цикл. Годы внутри цикла носили названия животных: крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, петуха, собаки и свиньи. Написать программу, которая по номеру года определяет его название в старояпонском календаре, если известно, что 1996 г. был годом крысы - началом очередного цикла.

***Решение.*** Поскольку цикл является двенадцатилетним, поставим в соответствие название года остатку от деления номера этого года на 12. При этом учтем, что остаток от деления 1996 на 12 равен 4.

```
Program Goroskop;

Var Year : Integer;

Begin

  Write('Введите год: '); ReadLn(Year);

  Case Year mod 12 Of

    0 : WriteLn('Год Обезьяны');

    1 : WriteLn('Год Петуха');

    2 : WriteLn('Год Собаки');

    3 : WriteLn('Год Свиньи');

    4 : WriteLn('Год Крысы');

    5 : WriteLn('Год Коровы');

    6 : WriteLn('Год Тигра');

    7 : WriteLn('Год Зайца');

    8 : WriteLn('Год Дракона');
```

```
    9 : WriteLn('Год Змеи');

    10 : WriteLn('Год Лошади');

    11 : WriteLn('Год Овцы');

  End;

End.
```

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Составить программу, решающую следующую задачу. Даны действительные числа $x$ и $y$, не равные друг другу. Меньшее из этих двух чисел заменить половиной их суммы, а большее – их удвоенным произведением.

***Задание 2.*** Составить программу, решающую следующую задачу. Даны два угла треугольника (в градусах). Определить, существует ли такой треугольник. Если да, то будет ли он прямоугольным.

***Задание 3.*** Составить программу, вычисляющую значение функции

$$F(x)=\begin{cases} x^2+4x+5, & \text{если } x \le 2 \\ \dfrac{1}{x^2+4x+5}, & \text{если } x > 2 \end{cases}.$$

***Задание 4.*** Составить программу, которая по данному числу (1-12) выводит название соответствующего ему месяца.

## 4 ПРАКТИЧЕСКАЯ РАБОТА № 3. Программирование циклических алгоритмов

***Тема:*** программирование циклических алгоритмов.

***Цель занятия:*** сформировать у обучаемых навыки программирования циклических алгоритмов.

***Основные понятия***

**Алгоритм** - точное предписание, определяющее вычислительный процесс, ведущий от варьируемых начальных данных к искомому результату.

**Алгоритм циклической структуры (повторение)** – алгоритмы, отдельные действия в которых многократно повторяются.

**Цикл** - совокупность действий алгоритма, связанную с повторением.

**Параметр цикла** - величина, с изменением которой связано многократное выполнение цикла.

**Язык программирования** - система обозначений для точного описания алгоритмов для ЭВМ.

**Программа** — текст, задающий множество процессов вычислений, в соответствии с которым исполнитель, понимающий программу, разворачивает какой-то один из них.

**Программирование** - процесс определения последовательности инструкций, которые должен выполнить компьютер для решения определенной задачи.

**Система программирования** - совокупность языка программирования и виртуальной машины, обеспечивающей выполнение реальной машиной программ, составленных на этом языке.

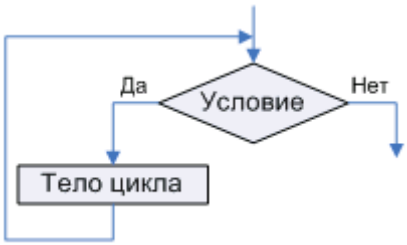
***Вводная часть***

*Цикл* - многократное повторение последовательности действий по некоторому условию. Известны три типа циклических алгоритмических структур: цикл с предусловием, цикл с постусловием и цикл с параметром. В Паскале существуют операторы, реализующие все три типа циклов.

*Цикл с предусловием* (цикл-пока) — наиболее универсальная циклическая структура. Реализуется оператором While. Формат оператора:

While <логическое выражение> Do <тело цикла>

Пока значение логического выражения — True, выполняется тело цикла. Тело цикла может быть простым или составным оператором.




*Цикл с постусловием* (цикл-до) имеет формат:

Repeat <тело цикла> Until <логическое выражение>

Повторяется выполнение тела цикла. Цикл заканчивается, когда логическое выражение принимает значение True. Тело цикла с постусловием выполняется хотя бы один раз. Использования Begin и End для ограничения составного тела цикла не требуется.


*Цикл с параметром* имеет два варианта записи:

1) For I := In To Ik Do <тело цикла>;
2) For I: = In DownTo Ik Do <тело цикла>.

Здесь I — параметр цикла - простая переменная порядкового типа;

In — выражение того же типа, определяющее начальное значение параметра;

Ik — выражение того же типа, определяющее конечное значение параметра;

<тело цикла> может быть простым или составным оператором.

Цикл повторяется, пока значение параметра лежит в интервале между In и Ik. Причем эти выражения вычисляются только один раз в начале выполнения цикла.

В первом варианте при каждом повторении цикла значение параметра изменяется на следующее значение в данном типе (для целого типа — увеличивается на 1). Во втором варианте при каждом повторении цикла значение параметра изменяется на предыдущее значение в данном типе (для целого типа -уменьшается на 1).

***Пример 1.*** Вычислить сумму натурального ряда чисел от 1 до N.

***Решение.*** Программа будет состоять из трех частей, в которых повторяется решение этой задачи с использованием операторов цикла While, Repeat и For.

```
Program Natur;
Var a,N,Summa : Integer;
Begin
  Write('N='); ReadLn(N);
  {Цикл с предусловием}
  a:=1; Summa:=0;
  While a<=N Do
  Begin
     Summa:=Summa+a;
     a:=a+1;
  End;
  WriteLn('Summa=',Summa);
  {Цикл с постусловием}
  a:=1; Summa:=0;
  Repeat
     Summa:=Summa+a;
     a:=a+1;
  Until a>N;
  WriteLn('Summa=',Summa);
  {Цикл с параметром}
  Summa:=0;
  For a:=1 To N Do Summa:=Summa+a;
  WriteLn('Summa=',Summa);
End.
```

Очевидно, что все три результата будут одинаковыми.

***Пример 2.*** Функцию $y = \sqrt{x}$ можно вычислить как предельное значение последовательности, определяемой рекуррентной формулой $y_k = \frac{y_{k-1} + \frac{x}{y_{k-1}}}{2}$ *для* $k = 1,2, \ldots$.

Начальное значение $y_0$ задается произвольно. За приближенное с точностью e значение корня берется первое $y_k$, для которого выполняется условие $|y_k - y_{k-1}| < \varepsilon$ .

***Решение.*** Для вычисления значений числовой последовательности достаточно двух простых переменных, в которых на каждом шаге будут храниться последнее и предпоследнее значения: $y_k$ и $y_{k-1}$. Обозначим эти

переменные Anew и Aold. При программировании этой задачи нельзя использовать цикл с параметром, т. к. неизвестно заранее число повторений цикла. Воспользуемся циклом с предусловием.

```
Program Posled;
Var X,eps,Aold,Anew : Real; k : Integer;
Begin
  Write('eps='); ReadLn(eps);
  Write('X='); ReadLn(X);
  Aold:=X; Anew:=(Aold+X/Aold)/2;
  While Abs(Anew-Aold)>=eps Do
  Begin
    Aold:=Anew;
    Anew:=(Aold+X/Aold)/2;
  End;
  WriteLn('Корень квадратный(',X,')= ',Anew);
End.
```

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Составить программу вычисления значений функции *F(x)=x-sin(x)* на отрезке [a;b] с шагом h.

***Задание 2.*** Составить программу, вычисляющую произведение N натуральных чисел.

***Задание 3.*** Составить программу, решающую следующую задачу. Дано натуральное число N. Вычислить $S = 1 - \frac{1}{2} + \frac{1}{4} - \frac{1}{8} + \cdots + (-1)^N \frac{1}{2^N}$.

## 5 ПРАКТИЧЕСКАЯ РАБОТА № 4. Программирование задач с использованием массивов

***Тема:*** программирование задач с использованием массивов.
***Цель занятия:*** сформировать у обучаемых навыки программирования с использованием массивов.

***Основные понятия***

**Алгоритм циклической структуры (повторение)** – алгоритм, отдельные действия в которых многократно повторяются.

**Цикл** - совокупность действий алгоритма, связанную с повторением.

**Параметр цикла** - величина, с изменением которой связано многократное выполнение цикла.

**Массив** - упорядоченный набор однотипных значений.

**Линейный (одномерный) массив** — массив, у которого элементы — простые переменные.

**Двумерный массив** - структура данных, хранящая прямоугольную матрицу.

***Вводная часть***

*Массив* — упорядоченный набор однотипных значений — компонент массива. Тип компонент называется базовым типом массива.

В Паскале массив рассматривается как переменная структурированного типа. Массиву присваивается имя, посредством которого можно ссылаться на него, как на единое целое, так и на любую из его компонент.

*Переменная с индексом* — идентификатор компоненты массива. Формат записи:

<имя массива>[<индекс>],

где индекс может быть выражением порядкового типа.

Описание массива определяет имя, размер массива и базовый тип. Формат описания в разделе переменных:

Var <имя массива > : Array [<тип индекса>] Of <базовый тип>;

Чаще всего в качестве типа индекса используется интервальный целый тип.

Линейный (одномерный) массив — массив, у которого элементы — простые переменные. В одномерных массивах хранятся значения линейных таблиц. Примеры описания одномерных массивов:

```
Var B : Array[0..5] Of Real;
    R : Array[1..34] Of Char;
    N : Array['A'..'Z' ] Of Integer;
```

*Ввод и вывод массива* производится поэлементно. Обычно для этого используется цикл с параметром, где в качестве параметра применяется индексная переменная.

***Пример 1.*** В программе вводится десять значений целочисленного массива A и выводятся значения вещественного массива B, содержащего 50 элементов. Соответствующие фрагменты программы:

```
Var A : Array[1..10] Of Integer;

  B : Array[1..50] Of Real;

  I : Integer;

Begin

 For I:=1 To 10 Do

 Begin
```

```
    write('A[',I,']='); readln(A[I]);

  End;

  ...

  For I:=1 To 50 Do

    writeln('B[',I,']=',B[I});

End.
```

***Пример 2.*** Заполнить числами из диапазона [0,1] вещественный линейный массив из N чисел. Найти максимальное значение и его индекс (первый, если таких значений несколько).

***Решение.*** Поскольку размер массива в программе должен быть однозначно задан, определим N в разделе констант, например N = 20. При изменении размера массива достаточно будет отредактировать в программе лишь описание константы N.

```
Const N=10;
Var X : Array[1..N] Of Real;
   I : Integer;
   max : Real; Kmax : Integer;
Begin
  For I:=1 To N Do
  Begin
    write('X[',I,']='); readln(X[I]);
  End;
  max:=X[1]; Kmax:=1;
  For I:=2 To N Do
  If X[I]>max Then
  Begin
    max:=X[I]; Kmax:=I;
  End;
  writeln('Первое максимальное значение: X[',Kmax,']=',max)
End.
```

***Пример 3.*** Дан целочисленный линейный массив. Отсортировать его элементы в порядке уменьшения значений.

***Решение.*** Воспользуемся алгоритмом, известным под названием «метод пузырька». Идея состоит в последовательном перемещении путем попарных перестановок наибольшего значения сначала на место M-го элемента, затем N-1 -го и т.д. Опишем массив на максимальный размер (например 100), а фактический размер N определим вводом.

```
Var A : Array[1..100] Of Integer;

  N,I,J,P : Integer;
```

```
    max : Real; Kmax : Integer;
  Begin
    write('N='); readln(N);
    For I:=1 To N Do
    Begin
      write('A[',I,']='); readln(A[I]);
    End;
    For I:=1 To N Do
    Begin
      For J:=1 To N-I Do
      If A[J]<=A[J+1] Then
      Begin
        P:=A[J]; A[J]:=A[J+1]; A[J+1]:=P;
      End;
    End;
    For I:=1 To N Do write(A[I], ' ')
  End.
```

Двумерный массив - структура данных, хранящая прямоугольную матрицу. В матрице каждый элемент определяется номером строки и номером столбца, на пересечении которых он расположен.

В Паскале двумерный массив рассматривается как массив, элементы которого есть линейные массивы. Два следующих описания двумерных массивов тождественны:

```
Var M : Array[1..10] Of Array[1..20] Of Real;
Var M : Array[1..10,1..20] Of Real;
```

Чаще пользуются вторым способом описания. Элементы двумерного массива идентифицируются переменными с двумя индексами. Например: M[3, 5]. Обычно первый индекс связывают с номером строки, второй — с номером столбца матрицы.

***Пример 4.*** Сформировать матрицу Пифагора (таблицу умножения в матричной форме) и вывести ее на экран.

***Решение.*** Значения элементов матрицы Пифагора вычисляются следующим образом: P[I,J]=I*J. Вычисления и вывод матрицы производятся в двух вложенных циклах. Вывод на экран организуем в виде прямоугольной таблицы.

```
Program Pifagor;
Var P : Array[1..9,1..9] Of Integer;
   I,J : Integer;
Begin
 For I:=1 To 9 Do
 For J:=1 To 9 Do
    P[I,J]:=I*J;
 For I:=1 To 9 Do
 Begin
  For J:=1 To 9 Do Write(P[I,J]:8);
  writeln;
 End;
End.
```

***Пример 5.*** Заполнить матрицу порядка *n* по следующему образцу:

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | ... | $n-2$ | $n-1$ | $n$ |
| 2 | 1 | 2 | ... | $n-3$ | $n-2$ | $n-1$ |
| 3 | 2 | 1 | ... | $n-4$ | $n-3$ | $n-2$ |
| ... | ... | ... | ... | ... | ... | ... |
| $n-1$ | $n-2$ | $n-3$ | ... | 2 | 1 | 2 |
| $n$ | $n-1$ | $n-2$ | ... | 3 | 2 | 1 |

***Решение.*** Идея алгоритма основана на двух свойствах этой матрицы: она симметрична относительно главной диагонали, т.е. A[I,J] = A[J,I], и элементы верхнего треугольника матрицы вычисляются по формуле A[I,J] = J - I + 1. Программа составлена для n=10.

```
Program Massiv;
Var A : Array[1..10,1..10] Of Integer;
   I,J,K,N : Integer;
Begin
 write('N='); readln(N);
 For I:=1 To N Do
 For J:=1 To N Do
 Begin
   A[I,J]:=J-I+1;
   A[J,I]:=A[I,J];
 End;
 For I:=1 To N Do
 Begin
  For J:=1 To N Do Write(A[I,J]:8);
  writeln;
 End;
End.
```

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Составить программу, решающую следующую задачу. Даны действительные числа $x_1, x_2, \ldots x_n$. Поменять местами наибольший и наименьший элементы.

***Задание 2.*** Составить программу, формирующую квадратную матрицу порядка *n* по заданному образцу:

$$\begin{matrix} 1 & 1 & 1 & \ldots & 1 & 1 & 1 \\ 1 & 0 & 0 & \ldots & 0 & 0 & 1 \\ 1 & 0 & 0 & \ldots & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & \ldots & 0 & 0 & 1 \\ 1 & 1 & 1 & \ldots & 1 & 1 & 1 \end{matrix}$$

***Задание 3.*** Составить программу, вычисляющую сумму и число положительных элементов матрицы A[N,N], находящихся над главной диагональю.

## 6 ПРАКТИЧЕСКАЯ РАБОТА № 5. Программирование задач с использованием подпрограмм

***Тема:*** программирование задач с использованием подпрограмм.

***Цель занятия:*** сформировать у обучаемых навыки программирования с использованием подпрограмм.

***Основные понятия***

**Вспомогательные алгоритмы (подалгоритмы)** - алгоритмы, используемые в составе других алгоритмов.

**Подпрограмма** — программа, реализующая вспомогательный алгоритм.

**Основная программа** — программа, реализующая основной алгоритм решения задачи и содержащая в себе обращения к подпрограммам.

**Фактические параметры** - реальные объекты программы, заменяющие в теле процедуры при ее вызове формальные параметры.

**Формальные параметры** — переменные, фиктивно (формально) присутствующие в процедуре и определяющие тип и место подстановки фактических параметров.

**Рекурсивная подпрограмма** – подпрограмма, которая в своем описании содержит обращение к самой себе.

***Вводная часть***

*Подпрограмма* — программа, реализующая вспомогательный алгоритм. *Основная программа* — программа, реализующая основной алгоритм решения задачи и содержащая в себе обращения к подпрограммам. В Паскале существуют два типа подпрограмм: подпрограммы-функции и подпрограммы-процедуры. Используемые в программе нестандартные подпрограммы должны быть описаны в разделе описания подпрограмм.

*Подпрограмма-функция* имеет следующий формат описания:

Function <имя функции> (<параметры-аргументы>) : <тип функции>;
<блок>;

Тип функции может быть только простым типом (в Турбо-Паскале исключением из этого правила является тип string). Блок содержит локальные для функции описания и раздел операторов. Обращение к функции является операндом в выражении.

*Подпрограмма-процедура* имеет следующий формат описания:

Procedure <имя процедуры> (<параметры>);
<блок>;

В качестве результата процедура может возвращать в вызывающую программу множество простых или структурированных величин или не возвращать никаких значений. Среди параметров процедуры указываются как аргументы, так и результаты. Параметры-результаты должны быть обязательно параметрами-переменными (описанными после служебного слова Var). Обращение к процедуре - отдельный оператор.

Обмен данными между вызывающей программой и подпрограммой может происходить не только через параметры, но и через глобальные переменные.

***Пример 1.*** Вычислить разность двух простых дробей $a/b - c/d$ ($a, b, c, d$ – натуральные числа). Результат получить в виде простой несократимой дроби $\frac{e}{f}$.

***Решение.*** Следует вычислить числитель и знаменатель по правилам вычитания дробей и сократить их на наибольший общий делитель (НОД). Вычисление НОД двух чисел оформим в виде подпрограммы, используя известный в математике алгоритм Евклида.

Составим два варианта программы решения этой задачи: с подпрограммой-функцией и подпрограммой-процедурой.

***Решение 1***

```
Program Sub1;
Var A,B,C,D,G,E,F : Integer;
Function Nod(M,N : Integer) : Integer;
Begin
  while (M<>N) Do
    If M>N Then M:=M-N Else N:=N-M;
  Nod:=M;
End;
Begin
  write('Введите числители и знаменатели дробей: '); readln(A,B,C,D);
  E:=A*D-B*C;
  F:=B*D;
  If E=0 Then writeln(E)
  Else
  Begin
    G:=Nod(Abs(E),F);
    E:=E div G;
    F:=F div G;
    writeln('Ответ: ',E,'/',F);
  End;
End.
```

***Решение 2***

```
Program Sub2;
Var A,B,C,D,G,E,F : Integer;
Procedure Nod(M,N : Integer; Var K : Integer);
Begin
  while (M<>N) Do
    If M>N Then M:=M-N Else N:=N-M;
```

```
  K:=M;
End;
Begin
  write('Введите числители и знаменатели дробей: '); readln(A,B,C,D);
  E:=A*D-B*C;
  F:=B*D;
  If E=0 Then writeln(E)
  Else
  Begin
    Nod(Abs(E),F,G);
    E:=E div G;
    F:=F div G;
    writeln('Ответ: ',E,'/',F);
  End;
End.
```

***Пример 2.*** Составить рекурсивную подпрограмму-функцию вычисления факториала целого положительного числа.

***Решение.*** Рекурсивной называется подпрограмма, которая в своем описании содержит обращение к самой себе. Функцию N! рекурсивно можно определить исходя из следующей формулы:

$$N! = \begin{cases} 1, & \text{если } N = 0 \\ (N-1)! \times N, & \text{если } N > 0 \end{cases}.$$

Описание функции на Паскале:

```
Function factorial(N : Integer) : Integer;
Begin
  If N=0 Then factorial:=1
         Else factorial:=factorial(N-1)*N;
End;
```

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Составить программу, возводящую положительное число в степень, используя подпрограмму-функцию: $x^y = e^{y \ln x}$.

***Задание 2.*** Составить программу, вычисляющую сумму элементов трех массивов. Ввод и расчет суммы элементов каждого массива организовать с помощью двух подпрограмм-процедур.

***Задание 3.*** Составить программу вычисления суммы факториалов всех нечетных чисел от 1 до 9.

## 7 ЛИТЕРАТУРА


**Основная учебная**

1. **Синицына, Т.Г.** Введение в программирование на языке С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

2. **Синицына, Т.Г.** Основы объектно-ориентированного программирования в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

3. **Синицына, Т.Г.** Программирование в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.

4. **Львович И.Я.** Основы информатики [Электронный ресурс]: учебное пособие/ Львович И.Я., Преображенский Ю.П., Ермолова В.В.— Электрон. текстовые данные.— Воронеж: Воронежский институт высоких технологий, 2014.— 339 c. http://www.iprbookshop.ru/23359.— ЭБС «IPRbooks»

5. **Фарафонов А.С.** Программирование на языке высокого уровня [Электронный ресурс]: методические указания к проведению лабораторных работ по курсу «Программирование»/ Фарафонов А.С.— Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013.— 32 c.— http://www.iprbookshop.ru/22912.— ЭБС «IPRbooks»

**Дополнительная**

1. **Глазырина, И.Б.** Введение в программирование. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

2. **Глазырина, И.Б.** Основные типы данных в Турбо Паскале. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

3. **Глазырина, И.Б.** Модульное программирование.[Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.

4. **Глазырина, И.Б.** Динамические структуры. [Электронный ресурс]: рабочий учебник/ Лабзина, Т.А. - 2011. - http://lib.muh.ru.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ
ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»
(для направления подготовки «Информатика
и вычислительная техника»)

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки Е.В. Белюсенко

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ) ПО ДИСЦИПЛИНЕ «ИНФОРМАТИКА (КУРС 7)»

МОСКВА 2018

Разработано Е.В. Корнеевой

Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ) ПО ДИСЦИПЛИНЕ «ИНФОРМАТИКА (КУРС 7)»

Методические указания подготовлены для обучающихся в образовательной организации и предназначены для изучения основных разделов информатики под общим наименованием «Профессиональные лабораторные занятия» (ПЛЗ) по направлению 09.03.01 «Информатика и ВТ» в рамках дисциплины «Информатика (курс 7)».

## I ОБЩИЕ ПОЛОЖЕНИЯ

**Цель ПЛЗ** заключается в формировании знаний в области теории информации, форм представления, обработки и передачи информации; изучении принципов построения информационных моделей и алгоритмизации, использовании технических и программных средств реализации информационных процессов, сетей ЭВМ.

**Особенность проведения** данного вида занятий заключается в последовательности осуществления практических и познавательных действий. Первыми выполняются мероприятия по заполнению регистрационных бланков. Это связано с тем, что большинство занятий дистанционного типа предполагает процесс самодиагностики и самопознания. Каждое занятие содержит теоретические сведения, практические задания, методику их выполнения, а также контрольные вопросы и упражнения для самостоятельной работы.

Каждое занятие подразделяется на следующие части:

1) *вступительная*. Обучаемые знакомятся с темой, целью, порядком проведения занятия, его значимостью для профессиональной деятельности, критериями оценки качества отработки заданий, рекомендациями по использованию учебной литературы;

2) *теоретическая*. Обучаемые самостоятельно изучают теоретические сведения по теме занятия, примеры, иллюстрирующие теоретические положения;

3) *практическая*. Обучаемые самостоятельно выполняют практические задания по теме занятия в соответствии с методикой их выполнения, представляют полученные результаты;

4) *тестирование.* Обучаемые самостоятельно отвечают на контрольные вопросы по теме занятия;

5) *самостоятельная работа.* Обучаемые выполняют упражнения для самостоятельной работы по вариантам, заполняют отчет, отвечая на поставленные вопросы;

6) *заключительная*. Предназначена для подведения итогов, контроля качества усвоения материала и оценки навыка использования изученных методов и средств обработки и представления информации. Подводятся итоги занятия, обучаемым выставляются оценки.

**Перечень профессиональных лабораторных занятий**

| | |
|---|---|
| **Лабораторное занятие № 1.**<br>Программирование машины Поста.<br><br>**Код занятия:**<br>1533.01.01;ЭПТ2.04;1 | Ознакомление с принципами функционирования машины Поста.<br>Приобретение знаний и навыков составления программ для машины Поста |
| **Лабораторное занятие № 2.** Позиционные системы счисления. Перевод целых чисел из десятичной системы счисления в другую и обратно.<br><br>**Код занятия:**<br>1533.01.01;ЭПТ2.01;1 | Ознакомление с позиционной системой счисления.<br>Приобретение знаний и навыков перевода целых десятичных чисел в двоичные, восьмеричные, шестнадцатеричные; целых двоичных, восьмеричных, шестнадцатеричных чисел в десятичные |
| **Лабораторное занятие № 3.** Позиционные системы счисления. Перевод действительных чисел из десятичной системы счисления в другую и обратно.<br><br>**Код занятия:**<br>1533.01.01;ЭПТ2.02;1 | Ознакомление с правилами перевода действительных десятичных чисел в двоичные, восьмеричные, шестнадцатеричные; действительных двоичных, восьмеричных, шестнадцатеричных чисел в десятичные.<br>Приобретение знаний и навыков перевода действительных десятичных чисел в двоичные, восьмеричные, шестнадцатеричные; действительных двоичных, восьмеричных, шестнадцатеричных чисел в десятичные |
| **Лабораторное занятие № 4.**<br>Знакомство с двоичной системой счисления. Овладение навыками перевода двоичных чисел в восьмеричную, шестнадцатеричную системы счисления и обратно. Знакомство с правилами сложения и умножения двоичных чисел. | Ознакомление с двоичной системой счисления, правилами двоичной арифметики.<br>Приобретение знаний и навыков перевода двоичных чисел в восьмеричную, шестнадцатеричную системы счисления и обратно; сложения и умножения двоичных чисел |

| | |
|---|---|
| **Код занятия:**<br>1533.01.01;ЭПТ2.03;1 | |
| **Лабораторное занятие № 5.**<br>Знакомство с формами представления двоичных чисел в ЭВМ.<br><br>**Код занятия:**<br>1533.02.01;ЭПТ2.02;1 | Ознакомление с формами представления двоичных чисел в ЭВМ.<br>Приобретение знаний и навыков представления чисел с фиксированной точкой и плавающей точкой в цифровом автомате |
| **Лабораторное занятие № 6.**<br>Знакомство со способами выполнения операции алгебраического сложения над двоичными числами в сумматорах различного типа.<br><br>**Код занятия:**<br>1533.02.01;ЭПТ2.03;1 | Ознакомление с сумматорами прямого, обратного и дополнительного кода.<br>Приобретение знаний и навыков выполнения операций алгебраического сложения двоичных чисел в сумматорах разного типа |
| **Лабораторное занятие № 7.**<br>Методы решения задач алгебры логики.<br><br>**Код занятия:**<br>1533.02.01;ЭПТ2.01;1 | Ознакомление с основными операциями над высказываниями, законами алгебры логики, примерами применения законов алгебры логики для решения логических задач.<br>Приобретение знаний и навыков решения логических задач |
| **Лабораторное занятие № 8.**<br>Знакомство с OpenOffice.org.Writer. Создание простых текстовых документов.<br><br>**Код занятия:**<br>1178.03.01;ЭПТ2.01;1 | Ознакомление с возможностями текстового процессора OpenOffice.org.Writer.<br>Приобретение знаний и навыков редактирования и форматирования текстов в OpenOffice.org.Writer |
| **Лабораторное занятие № 9.** Ознакомление с возможностями программы OpenOffice.org Calc по созданию электронных таблиц.<br><br>**Код занятия:**<br>1533.03.01;ЭПТ2.01;1 | Ознакомление с возможностями табличного процессора OpenOffice.org Calc.<br>Приобретение знаний и навыков создания электронных таблиц в OpenOffice.org Calc |
| **Лабораторное занятие № 10.** Ознакомление с возможностями программы OpenOffice.org Calc по обработке списков.<br><br>**Код занятия:**<br>1533.03.01;ЭПТ2.02;1 | Ознакомление с возможностями табличного процессора OpenOffice.org Calc по обработке списков.<br>Приобретение знаний и навыков создания и обработки списков в OpenOffice.org. Calc |
| **Лабораторное занятие № 11.**<br>Методы контроля работы цифрового автомата. Кодирование информации по методу четности-нечетности. Коды Хэмминга.<br><br>**Код занятия:**<br>1533.02.01;ЭПТ2.04;1 | Ознакомление с правилами кодирования информации по методу четности-нечетности, кодами Хэмминга.<br>Приобретение знаний и навыков кодирования информации методами четности и Хэмминга |

## II МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОГО ЗАНЯТИЯ

### а) Материально-техническое обеспечение

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

### Литература

**Основная учебная**

1 **Сальникова, Н.А**. Информатика [Электронный ресурс]: учебное пособие/ Сальникова Н.А.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2013.— 142 c.— http://www.iprbookshop.ru/11320.— ЭБС «IPRbooks».

2 **Львович И.Я.** Основы информатики [Электронный ресурс]: учебное пособие/ Львович И.Я., Преображенский Ю.П., Ермолова В.В.— Электрон. текстовые данные.— Воронеж: Воронежский институт высоких технологий, 2014.— 339 c. http://www.iprbookshop.ru/23359.— ЭБС «IPRbooks»

3 **Гураков, А.В**. Информатика [Электронный ресурс]: учебное пособие/ Гураков А.В., Лазичев А.А.— Электрон. текстовые данные.— Томск: Эль Контент, Томский государственный университет систем управления и радиоэлектроники, 2012.— 120 c.— http://www.iprbookshop.ru/13934.— ЭБС «IPRbooks».

**Дополнительная**

1 **Метелица, Н.Т**. Информатика [Электронный ресурс]: учебное пособие/ Метелица Н.Т., Орлова Е.В.— Электрон. текстовые данные.— Краснодар: Южный институт менеджмента, 2009.— 114 c.— http://www.iprbookshop.ru/9554.— ЭБС «IPRbooks».

2 **Мещеряков, П.С**. Прикладная информатика [Электронный ресурс]: учебное пособие/ Мещеряков П.С.— Электрон. текстовые данные.— Томск: Эль Контент, Томский государственный университет систем управления и радиоэлектроники, 2012.— 132 c.— http://www.iprbookshop.ru/13962.— ЭБС «IPRbooks».

3 **Белянина, Н.В.,** Корнеева, Е.В. Введение в информатику. [Электронный ресурс]: рабочий учебник/ Белянина, Н.В., Корнеева, Е.В. - 2012. - http://lib.muh.ru.

4 **Белянина, Н.В.,** Корнеева, Е.В., Лабзина, Т.А. Функциональная и структурная организация ЭВМ. [Электронный ресурс]: рабочий учебник/ Белянина, Н.В., Корнеева, Е.В., Лабзина, Т.А. - 2012. - http://lib.muh.ru.

5 **Белянина, Н.В.,** Лабзина, Т.А. Программное обеспечение. [Электронный ресурс]: рабочий учебник/ Белянина, Н.В., Лабзина, Т.А. - 2012. - http://lib.muh.ru.

6 **Белянина, Н.В.,** Глазырина, И.Б., Корнеева, Е.В. Компьютерные сети. [Электронный ресурс]: рабочий учебник/ Белянина, Н.В., Глазырина, И.Б., Корнеева, Е.В. - 2012. - http://lib.muh.ru.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
**ПО ПРОВЕДЕНИЮ ПРОФЕССИОНАЛЬНЫХ**
**ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ)**
**ПО ДИСЦИПЛИНЕ «ИНФОРМАТИКА (КУРС 7)»**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.Н. Горбатова
Оператор компьютерной верстки В.Г. Буцкая

**4191.01.01;МУ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ
ПО ДИСЦИПЛИНЕ «ИНЖЕНЕРНАЯ И КОМПЬЮТЕРНАЯ ГРАФИКА»
(для направления подготовки
09.03.01 «Информатика и вычислительная техника»)

**МОСКВА 2018**

Разработано Л.А. Букштынович

Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для обучающихся

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ

ПО ДИСЦИПЛИНЕ «ИНЖЕНЕРНАЯ И КОМПЬЮТЕРНАЯ ГРАФИКА»

(для направления подготовки

09.03.01 «Информатика и вычислительная техника»)

Методические указания способствуют пониманию специфики организации и технологии проведения лабораторных практикумов по изучению средств и технологии разработки графических объектов под общим наименованием «Лабораторный практикум» по направлению 09.03.01 «Информатика и ВТ» в рамках дисциплины «Инженерная и компьютерная графика».

**Цель лабораторных практикумов** заключается в формировании целостного представления об инженерной и компьютерной графике, принципах, методах и средствах работы с графикой, умений и навыков применения программного обеспечения инженерной и компьютерной графики в практической деятельности.

**Особенность проведения** данного вида занятий заключаются в последовательности осуществления практических и познавательных действий. Первыми выполняются мероприятия по заполнению регистрационных бланков. Это связано с тем, что большинство занятий дистанционного типа предполагают процесс самодиагностики и самопознания. Каждое занятие содержит теоретические сведения, практические задания, методику их выполнения, а также контрольные вопросы и упражнения для самостоятельной работы.

Каждое занятие подразделяется на следующие части.

**Первая – вступительная.** Предполагаются следующие действия преподавателя: объявление темы и уточнение ее значимости для профессиональной деятельности; разъяснение цели и задач; определение порядка проведения и критериев оценки качества отработки заданий; доведение рекомендаций по использованию учебной литературы.

**Вторая – теоретическая.** Обучаемые самостоятельно изучают теоретические сведения по теме занятия, примеры, иллюстрирующие теоретические положения. Преподаватель помогает обучающимся, отвечает на вопросы.

**Третья – практическая.** Обучаемые самостоятельно выполняют практические задания по теме занятия в соответствии с методикой их выполнения. Преподаватель помогает обучающимся выполнить все необходимые процедуры, представить полученные результаты.

**Четвертая – тестирование.** Обучаемые самостоятельно отвечают на контрольные вопросы по теме занятия.

**Пятая – самостоятельная работа.** Обучаемые выполняют упражнения для самостоятельной работы по вариантам, заполняют отчет, отвечая на поставленные вопросы.

**Шестая – заключительная.** Предназначена для подведения итогов, контроля качества усвоения материала и оценки навыка использования изученных методов, средств обработки и представления информации.

**Преподавателю необходимо**: подвести итог занятия и выставить оценки; ответить на вопросы обучаемых; уточнить время и аудиторию для проведения консультации и ликвидации текущих задолженностей.

**ПЕРЕЧЕНЬ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ**

| | |
|---|---|
| **Лабораторный практикум № 1.**<br>Геометрические построения простых объектов.<br>**Код занятия:**<br>4191.01.01;ЭПТЛ2.01;1 | Ознакомление с рабочей средой программы Компас-3D LT.<br>Приобретение знаний и навыков по работе с интерфейсом Компас 3D-LT.<br>Приобретение знаний и навыков по созданию двумерного документа |
| **Лабораторный практикум № 2.**<br>Построение чертежа механических деталей.<br>**Код занятия:**<br>4191.01.01;ЭПТЛ2.02;1 | Ознакомление с основными средствами системы Компас-3D-LT для построения чертежа механических деталей.<br>Приобретение знаний и навыков по применению инструментов панелей инструментов (основных и вспомогательных).<br>Приобретение знаний и навыков по построению фасок, штриховки чертежей |
| **Лабораторный практикум № 3.**<br>Нанесение размеров и подписей на чертеже механических деталей.<br>**Код занятия:**<br>4191.02.01.ЭПТЛ2.02;1 | Ознакомление с технологией нанесения размеров и надписей на чертеже.<br>Приобретение знаний и навыков нанесения линейных, угловых и радиальных размеров |
| **Лабораторный практикум № 4.**<br>Построение чертежа объемных сплошных объектов.<br>**Код занятия:**<br>4191.02.01;ЭПТЛ2.03;1 | Ознакомление с общими принципами твердотельного моделирования.<br>Приобретение знаний и навыков использования операций Выдавливание, Вращение для построения твердотельной модели.<br>Приобретение знаний и навыков использования операций Скругление, Отверстие |
| **Лабораторный практикум № 5.**<br>Создание рисунков с помощью графического редактора PAINT.<br>**Код занятия:**<br>1534.01.01;ЭПТЛ2.01;1 | Ознакомление с интерфейсом программы.<br>Приобретение знаний и навыков по созданию изображений с помощью инструментов программы;<br>Приобретение знаний и навыков по преобразованию изображений |
| **Лабораторный практикум № 6.**<br>Возможности редактора Open Office.org Draw<br>**Код занятия:**<br>1534.02.01;ЭПТЛ2.01;1 | Ознакомление с основными возможностями программы OpenOffice.org DRAW, интерфейсом программы.<br>Приобретение знаний и навыков рисования линий, простейших геометрических фигур;<br>Приобретение знаний и навыков по применению команд Фигуры – Объединить, Вычесть и Пересечь к нескольким объектам |

| | |
|---|---|
| **Лабораторный практикум № 7.**<br>Практическая работа с редактором OpenOffice.org Draw.<br><br>**Код занятия:**<br>1534.02.01;ЭПТЛ2.02;1 | Ознакомление с основными возможностями программы OpenOffice.org DRAW, создания блок-схем.<br>Приобретение знаний и навыков применения инструментов для создания схем алгоритмов.<br>Приобретение знаний и навыков применения инструментов для создания информационных диаграмм |
| **Лабораторное занятие № 8.**<br>Создание презентации с помощью программы OpenOffice.org Impress.<br><br>**Код занятия:**<br>1534.03.01;ЭПТЛ2.04;1 | Ознакомление с основными возможностями программы OpenOffice.org Impress.<br>Приобретение знаний и навыков для создания презентации.<br>Приобретение знаний и навыков демонстрации презентации |

## МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ УЧЕБНОГО ЗАНЯТИЯ

**а) Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**а) Информационное обеспечение.**

– Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы.
- тренинговые и тестирующие программы.
- интеллектуальные роботизированные системы оценки качества выполненных работ.

– Информационные и роботизированные системы, программные комплексы, программное обеспечение для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ПО «Комбат»;
- ПО «ЛиК»;
- ПК «КОП»;
- ИР «Каскад».

## ЛИТЕРАТУРА


**Основная учебная**

1. **Перемитина, Т.О**. Компьютерная графика [Электронный ресурс]: учебное пособие/ Перемитина Т.О.— Электрон. текстовые данные.— Томск: Эль Контент, Томский государственный университет систем управления и радиоэлектроники, 2012.— 144 c.— http://www.iprbookshop.ru/13940.— ЭБС «IPRbooks».
2. **Тельной В.И.** Начертательная геометрия [Электронный ресурс]: графические конспекты лекций. Учебное наглядное пособие/ Тельной В.И.— Электрон. текстовые данные.— М.: Московский государственный строительный университет, ЭБС АСВ, 2015.— 71 c.— http://www.iprbookshop.ru/30516.— ЭБС «IPRbooks»
3. **Золотарева Н.Л.** Инженерная графика [Электронный ресурс]: учебное пособие для студентов обучающихся по направлению «Землеустройство и кадастры» дневной и заочной формы обучения/ Золотарева Н.Л., Менченко Л.В.— Электрон. текстовые данные.— Воронеж: Воронежский государственный архитектурно-строительный университет, ЭБС АСВ, 2013.— 111 c.— http://www.iprbookshop.ru/22673.— ЭБС «IPRbooks»

**Дополнительная**

1 **Пятибратов, А.П., Шевченко, П.Г.** Основы начертательной геометрии. [Электронный ресурс]: рабочий учебник/ Пятибратов, А.П., Шевченко, П.Г. - 2009. - http://lib.muh.ru

2 **Шевченко, П.Г**. Техническое черчение. [Электронный ресурс]: рабочий учебник/ Шевченко, П.Г. - 2010. - http://lib.muh.ru

3 **Шевченко, П.Г., Пятибратов, А.П**. Общие сведения о компьютерной графике. [Электронный ресурс]: рабочий учебник/ Шевченко, П.Г., Пятибратов, А.П. - 2009. - http://lib.muh.ru

4 **Шевченко, П.Г.** Компьютерная графика и геометрическое моделирование. [Электронный ресурс]: рабочий учебник/ Шевченко, П.Г. - 2009. - http://lib.muh.ru

5 **Пятибратов, А.П., Шевченко, П.Г.** Современные графические системы. [Электронный ресурс]: рабочий учебник/ Пятибратов, А.П., Шевченко, П.Г. - 2009. - http://lib.muh.ru

6 **Федянова, Н.А.** Инженерная графика [Электронный ресурс]: учебное пособие/ Федянова Н.А.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2009.— 150 c.— http://www.iprbookshop.ru/11317.— ЭБС «IPRbooks»

7 **Машихина, Т.П.** Компьютерная графика [Электронный ресурс]: учебное пособие/ Машихина Т.П.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2009.— 146 c.— http://www.iprbookshop.ru/11328.— ЭБС «IPRbooks»

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ**
**ПО ДИСЦИПЛИНЕ «ИНЖЕНЕРНАЯ И КОМПЬЮТЕРНАЯ ГРАФИКА»**

(для направления подготовки
«Информатика и вычислительная техника»)

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки В.Г. Буцкая

**4308.01.01;МУ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

ПО ДИСЦИПЛИНЕ «ПРИНЦИПЫ ПОСТРОЕНИЯ WEB-СЕРВЕРОВ»

(для направления подготовки 09.03.01 «Информатика

и вычислительная техника»)

**МОСКВА 2018**

Разработано Е.В. Корнеевой
Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических пособий
для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «ПРИНЦИПЫ ПОСТРОЕНИЯ WEB-СЕРВЕРОВ»

(для направления подготовки 09.03.01 «Информатика
и вычислительная техника»)

Методические указания по проведению практических занятий по дисциплине «Принципы построения Web-серверов», направление 09.03.01 «Информатика и вычислительная техника», предназначены для закрепления и дополнения знаний, полученных на лекциях и практических занятиях.

**О Г Л А В Л Е Н И Е**

Стр.

**Введение**

Методические материалы представляют собой комплекс лабораторных практикумов для аудиторной работы, а также указаний и разъяснений, позволяющих обучающимся сформировать знания в области применения языка разметки HTML.

Настоящие методические указания по выполнению практических занятий по курсу «Принципы построения Web-серверов» составлены на основе требований Федерального Государственного образовательного стандарта высшего образования .

Основные задачи практических заданий направлены на то, чтобы познакомить обучающихся:

- с конструкциями и командами разметки языка HTML;
- со средствами управления структурой содержимого Web-страниц;
- со способами представления графических изображений на Web-страницах.

В практических работах оцениваются владение знаниями в области применения языка разметки HTML.

## ПРАКТИЧЕСКАЯ РАБОТА № 1. РАЗМЕЩЕНИЕ И ОФОРМЛЕНИЕ ТЕКСТА НА WEB-СТРАНИЦАХ

***Тема:*** размещение и оформление текста на Web-страницах.

***Цель занятия:*** сформировать у обучаемых навыки размещения и оформления текста на Web-страницах.

***Основные понятия***

**Web-сайт** — совокупность Web-страниц с повторяющимся дизайном (не обязательно), объединенных по смыслу, навигационно и физически находящихся на одном Web-сервере.

**Web-сервер** — сервер, хранящий и предоставляющий во внешнюю сеть данные, организованные в виде Web–страниц.

**Язык гипертекстовой разметки HTML (HyperText Markup Language)** — стандартный язык, предназначенный для создания гипертекстовых документов в среде WWW.

**Гипертекст -** текст, представленный в виде ассоциативно связанных автономных блоков.

**Тэг** – код (набор символов), идентифицирующий некоторый элемент документа и обозначающий способ отображения этого элемента.

**Метатэги** – тэги, начинающиеся со слова META, размещаемые внутри блока HEAD и служащие для указания различных дополнительных сведений о содержимом Web-страницы.

**Браузер** - специальная программа, предназначенная для просмотра HTML-документов.

***Вводная часть***

*Структура Web-страницы.* Большая часть тэгов образует контейнер, состоящий из открывающего и закрывающего тэгов. Тэги можно набирать как заглавными, так и строчными буквами.

Web-страница помещается в контейнер <HTML>…</HTML> и состоит из двух частей: заголовка и отображаемого в браузере содержания.

Заголовок страницы помещается в контейнер <HEAD>…</HEAD>. Заголовок содержит название страницы, которое помещается в контейнер <TITLE>…</TITLE> и при просмотре отображается в верхней строке окна браузера.

Также в заголовок помещаются не отображаемые при просмотре метатэги, задающие кодировку страницы для ее правильного отображения в браузере, а также содержащие описание и ключевые слова страницы, которые в первую очередь просматривают роботы поисковых систем.

Отображаемое в браузере содержание страницы помещается в контейнер <BODY>…</BODY>.

***Пример 1***

```
<HTML>
<HEAD>
<TITLE>Знакомство с тэгами HTML</ TITLE >
</HEAD>
<BODY>
…
</BODY>
</HTML>
```

Для того, чтобы создать HTML, документ, необходимо:

- открыть любой текстовый редактор (например, блокнот, встроенный в Windows);
- набрать произвольный текст и разметить его HTML-тэгами;
- сохранить файл с расширением .htm или .html.

Теперь, если открыть созданный файл с помощью Web-браузера, он будет отображен как Web-страница.

*Заголовки* в HTML определяются тэгами <H1> - <H6> (H1 определяет самый крупный заголовок, а H6, самый мелкий).

***Пример 2***

```
<H1>Заголовок первого уровня</H1>
<H2>Заголовок второго уровня</H2>
<H3>Заголовок третьего уровня</H3>
<H4>Заголовок четвертого уровня</H4>
<H5>Заголовок пятого уровня</H5>
<H6>Заголовок шестого уровня</H6>
```

С помощью HTML-тэга <P> можно определить *абзац*. Абзацы используются для логической группировки текста. Перед и после текста абзаца браузер автоматически отступает одну строку.

***Пример 3***

```
<P>Это абзац</P>
<P>Это другой абзац</P>
```

В более простом случае, когда не требуется указания форматирования абзацев (выравнивания и пр.), допускается просто записать в конце каждого абзаца одиночный тэг <P>, который воспринимается уже не как «открывающий» тэг контейнера, а как команда на однократное выполнение действия по переходу на другой абзац.

Если же требуется выполнить принудительный перенос текста на новую строку (что визуально воспринимается как начало нового абзаца, но без автоматического создания браузером межабзацного промежутка), то в требуемом месте текста необходимо вставить одиночный тэг <BR>.

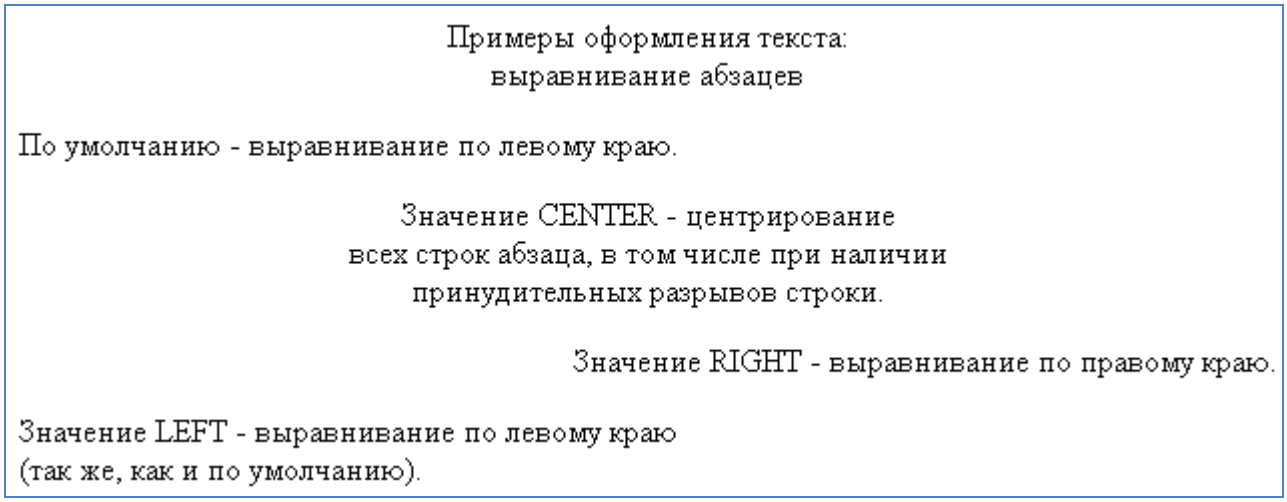
Одним из вариантов отделения текста может являться вставка горизонтальной линии с помощью HTML-тэга <HR>

По умолчанию абзацы выравниваются по левому краю. Требуемое выравнивание любого абзаца можно задать с помощью параметра ALIGN, записав его в открывающем тэге контейнера абзаца: <P ALIGN=…>, где вместо многоточия записывается значение CENTER (центрировать), RIGHT (выровнять по правому краю),

LEFT (выровнять по левому краю; именно это значение подразумевается по умолчанию) или JUSTIFY — выравнивание по ширине (по левому и правому краям одновременно). При использовании данного параметра в конце абзаца необходимо записать закрывающий тэг </P>.

***Пример 4***

```
<HTML>
<HEAD>
<TITLE>Примеры оформления текста: выравнивание абзацев</TITLE>
</HEAD>
<BODY>
<P ALIGN=CENTER>Примеры оформления текста: <BR>выравнивание абзацев</P>
<P>По умолчанию - выравнивание по левому краю.</P>
<P ALIGN=CENTER>Значение CENTER - центрирование <BR>всех строк абзаца,
в том числе при наличии <BR>принудительных разрывов строки.</P>
<P ALIGN=RIGHT>Значение RIGHT - выравнивание по правому краю.</P>
<P ALIGN=LEFT>Значение LEFT - выравнивание по левому краю<BR>
(так же, как и по умолчанию).</P>
</BODY>
</HTML>
```

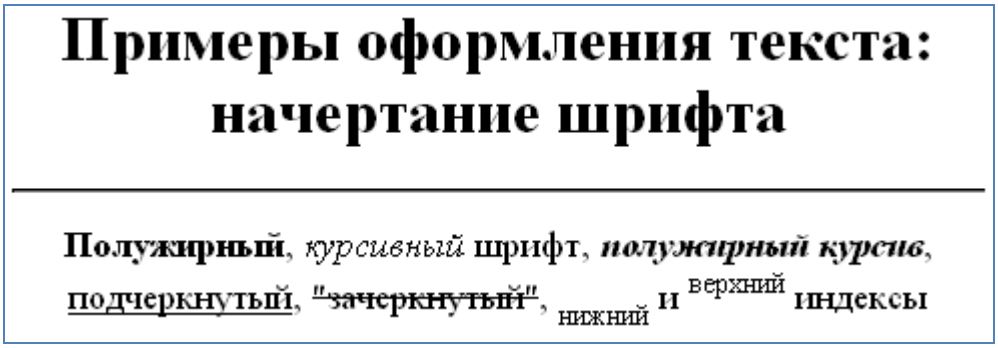



*Форматирование шрифта.* Для отдельных слов или фрагментов текста произвольной длины (в том числе в несколько абзацев) можно задать желаемое начертание - полужирное, курсивное или полужирный курсив, используя контейнеры <B>…</B> и <I>…</I> соответственно. Аналогичным образом можно указать подчеркнутое и «зачеркнутое» отображение текста при помощи контейнеров <U>…</U> и <S>…</S>, однако они используются гораздо реже, а кроме того, использование подчеркивания в HTML-документе нежелательно, чтобы пользователь не путал такой текст с гиперссылками. Эти контейнеры могут быть как вложены в контейнер абзаца, так и, наоборот, содержать в себе один или несколько абзацев. Аналогичным образом используются контейнеры <SUB>...</SUB> (для получения нижних индексов) и <SUP>…</SUP> (для верхних индексов).

***Пример 5***

```
<HTML>
<HEAD>
<TITLE>Примеры оформления текста: начертание шрифта</TITLE>
</HEAD>
<BODY>
<H1 ALIGN=CENTER>Примеры оформления текста: <BR>начертание шрифта</H1>
```

```
<HR SIZE=2>
<P ALIGN=CENTER>
<FONT SIZE=4><B>Полужирный</B>, <I>курсивный</I> шрифт,
<B><I>полужирный курсив</B></I>, <BR> <U>подчеркнутый</U>,
<S>"зачеркнутый"</S>, <SUB>нижний</SUB> и <SUP>верхний</SUP> индексы
</FONT></P>
</BODY>
</HTML>
```




*Списком* называется взаимосвязанный набор данных, которые начинаются с маркера или цифры. В HTML используется три типа списков:

- маркированный (или неупорядоченный) список,
- нумерованный (или упорядоченный) список,
- список определений.

Маркированный список определяется с помощью контейнера <UL>. Каждый элемент списка должен начинаться с элемента LI и быть вложенным в элемент UL.

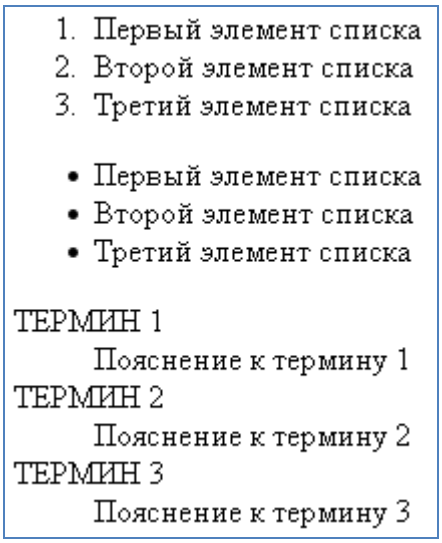
Нумерованный список устанавливается с помощью контейнера <OL>. Каждый элемент списка должен начинаться с элемента LI, как и в случае маркированного списка.

Для создания списка определений используются три элемента: DL, DT и DD. Каждый такой список начинается с контейнера <DL>, куда входит тэг <DT>, создающий термин, и тэг <DD>, задающий определение этого термина. Закрывающий тэг </DT> не обязателен, поскольку следующий тэг сообщает о завершении предыдущего элемента.

***Пример 6***

```
<OL>
<LI>Первый элемент списка</LI>
<LI>Второй элемент списка</LI>
<LI>Третий элемент списка</LI>
</OL>
<UL>
<LI>Первый элемент списка</LI>
<LI>Второй элемент списка<LI>
<LI>Третий элемент списка</LI>
<UL>
<DL>
<DT>ТЕРМИН 1</DT>
<DD>Пояснение к термину 1</DD>
```

```
<DT>ТЕРМИН 2</DT>
<DD>Пояснение к термину 2</DD>
<DT>ТЕРМИН 3</DT>
<DD>Пояснение к термину 3</DD>
</DL>
```




*Атрибуты задания цветовой схемы (цвета фона, текста и т.д.).* Цвет на Web-странице задают либо его названием, либо числовым шестиразрядным шестнадцатеричным кодом #RRGGBB (первые два разряда задают интенсивность красного цвета, вторые — зеленого и третьи — синего).

Значение яркости цвета может меняться от минимальной 00 до максимальной FF. В таблице приведены примеры некоторых цветов:

| цвет | код | название | цвет | код | название |
|---|---|---|---|---|---|
| черный | #000000 | black | фиолетовый | #FF00FF | magenta |
| белый | #FFFFFF | white | бирюзовый | #00FFFF | cyan |
| красный | #FF0000 | red | желтый | #FFFF00 | yellow |
| зеленый | #00FF00 | lime | золотой | #FFD800 | gold |
| синий | #0000FF | blue | оранжевый | #FFA500 | orange |
| серый | #808080 | gray | коричневый | #A82828 | brown |

Основную цветовую схему Web-страницы можно задать в тэге <BODY> с помощью атрибутов:

| | |
|---|---|
| Цвет фона | BGCOLOR="#RRGGBB" |
| Текстура фона | BACKGROUND="file_name" |
| Цвет текста | TEXT="#RRGGBB" |
| Цвет текста ссылки | LINK="#RRGGBB" |
| Цвет текста активной ссылки | ALINK="#RRGGBB" |
| Цвет текста просмотренной ссылки | VLINK="#RRGGBB" |

При использовании текстуры, закрывающей собой всю площадь страницы, применение однотонного фона кажется излишним. Однако рисунки загружаются несколько медленнее, чем текст. Все это время посетители страницы будут видеть цвет фона, заданный атрибутом BGCOLOR. Поэтому для фона указывают цвет, совпадающий с основным тоном фонового рисунка.

Чтобы текст хорошо читался, цвета на странице подбирают контрастирующие по яркости: пастельный фон — темный текст или темный фон — светлый текст. Нежелательны буквы белого цвета — они могут оказаться невидимыми при печати страницы на принтере.

***Пример 7***

<BODY BGCOLOR="#FFFFCC" BACKGROUND="fon.png"
TEXT="#993300" LINK="#OOFFOO" ALINK="#FFOOOO" VLINK="#OOFFOO">

*Мета-тэги.* В раздел заголовка Web-страницы могут быть добавлены информационные одиночные тэги <META>, имеющие атрибуты NAME, HTTP-EQUIV и CONTENT.

Мета-тэг может информировать браузер о кодировке Web-страницы:

<meta http-equiv= "Content-Type" content="text/html; charset=windows-1251">

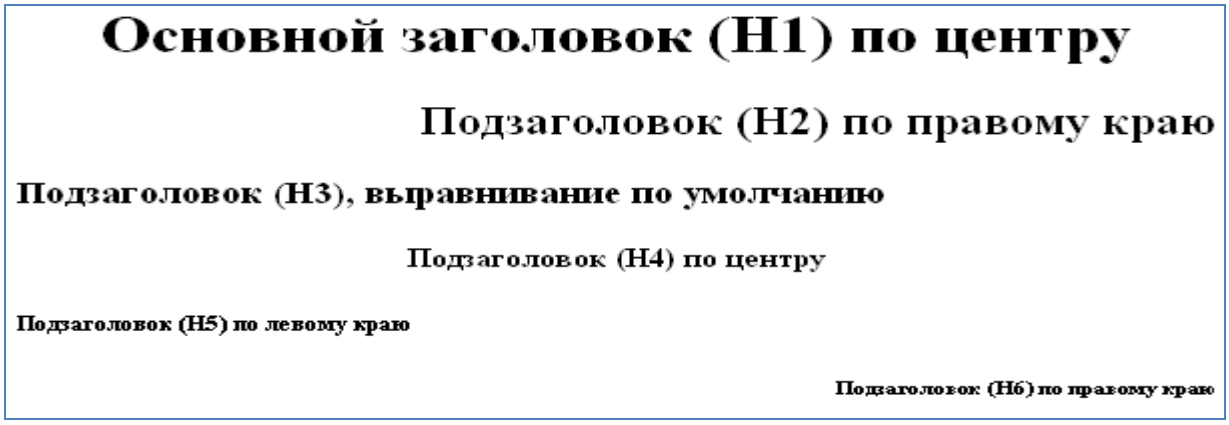
Мета-тэги используются поисковыми системами для индексирования содержания, ключевых слов и автора Web-страницы:

<meta name="Description" content="">
<meta name="Keywords" content="">
<meta name="Author" content="">

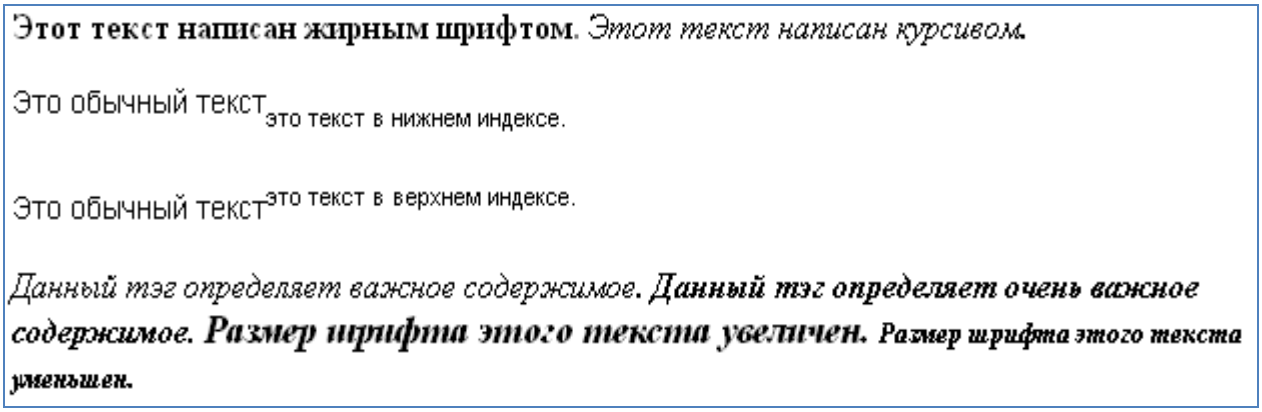
***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Создать HTML-документ, результат работы которого представлен на рисунке:




***Задание 2.*** Опишите тэги, используемые для формирования HTML-документа:




***Задание 3.*** Опишите результат работы HTML-документа:

<html>

```
<head>
<title>Примеры оформления текста: использование тэга &lt;FONT&gt;</title>
</head>
<body>
<h1 align=center>Примеры оформления текста:<br>
использование тэга &lt;FONT&gt;</h1>
<hr size=2>
<p align=center>
<font size=1>Ш</font><font size=+1>Р</font>
<font size=+2>И</font><font size=+1>Ф</font><font size=1>Т </font>
<font size=2>Р</font><font size=+1>А</font>
<font size=+2>З</font><font size=+3>Н</font>
<font size=+2>О</font><font size=+1>Г</font><font size=2>О </font>
<font size=3>Р</font><font size=+1>А</font>
<font size=+2>З</font><font size=+3>М</font>
<font size=+2>Е</font></font><font size=+1>Р</font><font size=3>А </font>
<font size=3>-- пример действия тэга &lt;FONT SIZE...&gt;</font>
</p>
<p align=center>
<font size=4 color=red>ШРИФТ </font>
<font size=4 color=green>РАЗНОГО </font>
<font size=4 color=blue>ЦВЕТА </font>
<font size=3>-- пример действия тэга &lt;FONT COLOR...&gt;</font>
</p>
<p align=center>
<font size=4 face="Arial Cyr">ШРИФТ </font>
<font size=4 face="Times New Roman Cyr">РАЗНОГО </font>
<font size=4 face="Courier New Cyr">НАЧЕРТАНИЯ</font>
<font size=3>-- пример действия тэга &lt;FONT FACE...&gt;</font>
</p>
<hr size=2>
</body>
</html>
```

***Задание 4.*** Создать HTML-документ, результат работы которого представлен на рисунке:

**Пример списка определений**

Анимация
: Создание иллюзии движения объекта на экране монитора за счет быстрой смены кадров, изображающих последовательные фазы движения.

Архиватор
: Программа, выполняющая сжатие (архивацию) файлов для уменьшения занимаемого ими дискового пространства и их распаковку (разархивацию) их в исходное состояние.

Блок-схема
: Графическое изображение алгоритма в виде соединенных линиями (стрелками) геометрических фигур.

***Задание 5.*** Опишите назначение метатэгов:

1) <meta name="description" content="Бесплатные онлайн учебники по HTML, CSS, JavaScript, jQuery, HDOM и AJAX"/>

2) <meta name="keywords" content="HTML, XHTML, CSS, JavaScript, jQuery, HDOM, AJAX"/>

---

## ПРАКТИЧЕСКАЯ РАБОТА № 2. ИСПОЛЬЗОВАНИЕ ИЗОБРАЖЕНИЙ И ГИПЕРССЫЛОК НА WEB-страницах

***Тема:*** использование изображений и гиперссылок на Web-страницах.

***Цель занятия:*** сформировать у обучаемых навыки использования изображений и гиперссылок на Web-страницах.

***Основные понятия***

**Web-сайт** — совокупность Web-страниц с повторяющимся дизайном (не обязательно), объединенных по смыслу, навигационно и физически находящихся на одном Web-сервере.

**Web-сервер** — сервер, хранящий и предоставляющий во внешнюю сеть данные, организованные в виде Web–страниц.

**Язык гипертекстовой разметки HTML (HyperText Markup Language)** — стандартный язык, предназначенный для создания гипертекстовых документов в среде WWW.

**Тэг** – код (набор символов), идентифицирующий некоторый элемент документа и обозначающий способ отображения этого элемента.

**Браузер** - специальная программа, предназначенная для просмотра HTML-документов.

**Гипертекст -** текст, представленный в виде ассоциативно связанных автономных блоков.

**Гиперссылка -** средство указания смысловой связи фрагмента одного документа с другим документом или его фрагментом.

***Вводная часть***

*Вставка изображений.* Кроме текста, на Web-странице можно разместить графические иллюстрации — созданные в графическом редакторе или оцифрованные с помощью сканера с фотографии, либо нарисованного от руки наброска, как цветные, так и черно-белые.

Большинство браузеров поддерживает только три графических формата: GIF, JPEG и PNG. Наиболее широкое распространение получили GIF и JPEG. Долгое время они являлись практически стандартами Web-изображений.

Для вставки изображения в HTML-документ используется тэг <IMG>, имеющий два обязательных параметра SRC и ALT. Атрибут ALT содержит так называемый альтернативный текст, который будет отображаться, если по каким-либо причинам изображение недоступно. Атрибут SRC задает адрес графического файла, который будет отображаться на Web-странице. В качестве значения принимается абсолютный или относительный адрес файла. К абсолютному адресу относится полный путь к ресурсу, включая протокол передачи данных, наименование сервера, а также имена всех каталогов, ведущих к файлу, например http://www.somewhere.com/images/image.jpg. Относительные адреса описывают местоположение файла относительно текущего каталога (например, "../../images/image.jpg") или корня каталогов сервера (например, "/images/image.jpg"). Если требуемый графический файл находится в том же каталоге, что и HTML-документ, его использующий, то в качестве значения аргумента SRC допустимо указать просто имя требуемого графического файла.

Можно управлять размерами выводимого на экран рисунка (независимо от его реальных размеров), используя параметр WIDTH=[ширина], где ширина картинки задается в процентах от ширины всей страницы (тогда после числа ставится знак «% ») или в пикселях. Например:

<IMG SRC="winter.gif" ALT="Зима в лесу" WIDTH=100>

Высота рисунка при этом изменяется пропорционально уменьшению/увеличению его ширины. Аналогично действует и еще один предусмотренный в HTML параметр HEIGHT, позволяющий указать высоту рисунка в пикселях, а при одновременном использовании WIDTH и HEIGHT можно указать точные размеры картинки, в том числе выполняя ее непропорциональное масштабирование — «растягивая» и «сплющивая» изображение.

Для улучшения внешнего вида страницы с иллюстрациями дополнительно можно использовать в тэге <IMG> параметры BORDER (рамка черного цвета вокруг иллюстрации, в качестве значения параметра указывается толщина в пикселях), HSPACE (поля слева и справа от рисунка) и VSPACE (поля сверху и снизу от рисунка). В качестве значений двух последних параметров также указываются размеры в пикселях.

*Различные виды гиперссылок.* Связать Web-страницу с другими документами можно с помощью универсального тэга <A> и его атрибута HREF, указывающих, в каком файле хранится вызываемый ресурс:

```
<A HREF="file_name">Указатель ссылки</A>
```

где file_name — путь к файлу или его URL-адрес в Интернете. Если вызываемый документ размещается в той же папке, что и Web-страница, то можно указывать только имя файла.

Указатель ссылки в окне браузера выделяется подчеркиванием и особым цветом. При указании на него мышью ее курсор превращается в значок «рука». Щелчок мыши по указателю вызывает переход на документ, указанный в гиперссылке.

Рассмотрим значения атрибута HREF для реализации различных реакций браузера:

Ссылка на Web-страницу локального компьютера:

```
<A HREF="clock.htm">Куранты</A>
```

Ссылка на Web-страницу, размещенную в Интернете:

```
<A HREF="http://www.moskva.ru">Сайт о Москве</A>
```

Вставка изображения:

```
<A HREF="spassk.jpg">Башня</A>
```

Запуск проигрывателя звукового файла:

```
<A HREF="strike.wav">Бой часов</A>
```

Сохранение файла на локальном компьютере:

```
<A HREF="kremlin.zip">Скачать файл</A>
```

Создание бланка письма с заполненным адресом получателя:

```
<A HREF="mailto:your name@freemail.ru">Создать письмо</A>
```

*Якоря. Гиперссылки на якоря.* Если создаваемая страница очень большая, то в ее тексте можно расставить якоря (иначе их еще называют метки или закладки). Они помечают смысловые разделы и помогают быстро перемещаться по документу.

Поместить метку top в начало страницы. Для этого атрибуту NAME тэга необходимо присвоить определенное значение (имя метки):

```
<A NAME="top"></A>
```

В конец страницы поместить указатель ссылки на метку. Для этого атрибуту HREF тэга необходимо присвоить значение (имя метки):

```
<A HREF="#top">На начало страницы</A>
```

Теперь, находясь в конце страницы, для перемещения в начало страницы достаточно осуществить щелчок по указателю гиперссылки *На начало страницы.*

Переключаться на якорь по гиперссылке можно не только внутри текущей Web-страницы, но и с другой Web-страницы.

*Ссылки с использованием карт-изображений (Map Image).* Для создания гиперссылок можно использовать карты-изображения, на которых выделены области — указатели гиперссылок. С каждой областью изображения связывают переход на определенный объект (внешнюю Web-страницу или внутренний якорь).

В качестве основы для создания карты-изображения можно выбрать любое изображение. Чтобы преобразовать изображение в карту, в тэг <IMG> добавляется атрибут USEMAP, значением которого является ссылка на имя описания конфигурации областей на карте. Для рисования рамки вокруг карты-изображения используется атрибут BORDER, значением которого является толщина рамки в пикселях:

```
<IMG SRC="cxema.png" WIDTH=265 HEIGHT=330 BORDER=1 USEMAP="#Kremlin"
ALIGN=left ALT="Схема Московского Кремля">
```

Для описания конфигурации областей карты-изображения используется тэг <MAP>, единственным атрибутом которого является NAME. Значение атрибута NAME должно соответствовать значению атрибута USEMAP, ранее заданному в тэге <IMG>:

```
<MAP NAME="Kremlin">
</MAP>
```

Форма и геометрические размеры областей на карте-изображении, а также адрес гиперссылки задаются тэгами <AREA> с атрибутами. Атрибут SHAPE определяет форму области и может принимать значения:

| Форма | Атрибуты |
|---|---|
| **Прямоугольная область-ссылка** | shape="rect", coords="x1,y1,x2,y2", где x1,y1 координаты верхнего левого угла прямоугольной области, а x2,y2 координаты нижнего правого угла области. |
| **Круглая область-ссылка** | shape="circle", coords="x,y,радиус", где x,y координаты центра области. |
| **Область-ссылка многоугольник** | shape="poly", coords="x1,y1,x2,y2..xz,yz", где x1,y2 координаты первого угла многоугольника, x2,y2 второго и т.д. Если координаты первого и последнего угла не совпадают, браузер автоматически добавит координату последнего угла, чтобы завершить многоугольник. |

Для задания адреса гиперссылки используется атрибут HREF, а для вывода альтернативного текста атрибут ALT:

```
<AREA SHAPE="rect" COORDS="175, 95,183, 119" HREF="#anchor9" ALT="Сенатская">
<AREA SHAPE="rect" COORDS="129, 21,148, 87" HREF="#anchor2" ALT="Никольская">
<AREA SHAPE="rect" COORDS="103, 3,116, 57" HREF="#anchor3"
ALT="Угловая   Арсенальная">
```

Координаты областей можно узнать, открыв рисунок в каком-либо графическом редакторе.

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Создать HTML-документ, результат работы которого представлен на рисунке:

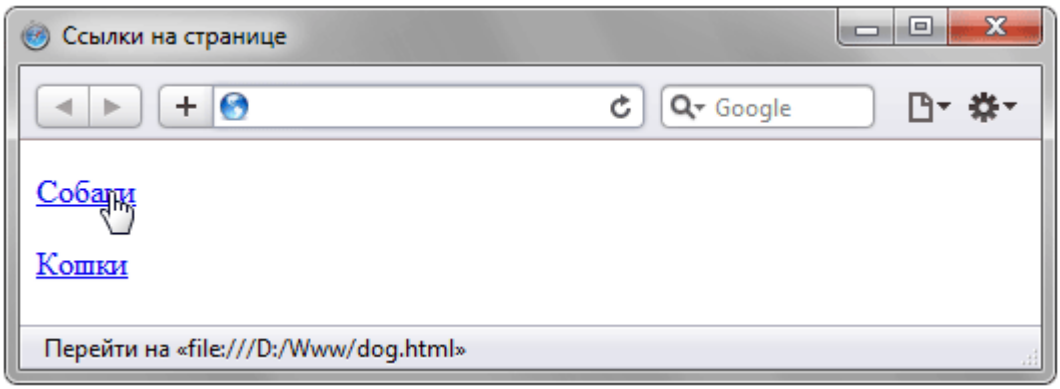

***Задание 2.*** Опишите результат работы HTML-документа:

```
<html>
<head>
<title>Пример вставки небольшой иллюстрации</title>
</head>
<body>
<h2 align=center>Способы вертикального выравнивания</h2>
<P>Математический символ <IMG SRC="Zero.jpg" ALIGN=TOP> означает, что данное множество является пустым.</P>
<P>Математический символ <IMG SRC="Zero.jpg" ALIGN=MIDDLE> означает, что данное множество является пустым.</P>
<P>Математический символ <IMG SRC="Zero.jpg" ALIGN=BOTTOM> означает, что данное множество является пустым.</P>
</body>
</html>
```

***Задание 3.*** Опишите результат работы фрагмента HTML-документа:

```
<a href="http://www.wisdomweb.ru"> www.wisdomweb.ru </a>
<img src='http://www.wisdomweb.ru/images/logo14.gif' width='200' height='60' /></a>
```

***Задание 4.*** Создать HTML-документ, результат работы которого представлен на рисунке:




***Задание 5.*** Опишите результат работы фрагмента HTML-документа:

```
<img src="figures.gif" usemap="#figuremap" />
<map name="figuremap">
<area shape="rect" coords="3,8,72,75" href="bluesquare.html" target="_blank" />
<area shape="circle" coords="125,50,38" href="greencircle.html" target="_blank" />
<area shape='poly' coords='180,27,210,6,248,30,233,74,190,72'
href="redpolygon.html" target="_blank" />
</map>
```

## ПРАКТИЧЕСКАЯ РАБОТА № 3. ИСПОЛЬЗОВАНИЕ ТАБЛИЦ НА WEB-страницах

***Тема:*** использование таблиц на Web-страницах.

***Цель занятия:*** сформировать у обучаемых навыки использования таблиц на Web-страницах.

***Основные понятия***

**Web-сайт** — совокупность Web-страниц с повторяющимся дизайном (не обязательно), объединенных по смыслу, навигационно и физически находящихся на одном Web-сервере.

**Web-сервер** — сервер, хранящий и предоставляющий во внешнюю сеть данные, организованные в виде Web–страниц.

**Язык гипертекстовой разметки HTML (HyperText Markup Language)** — стандартный язык, предназначенный для создания гипертекстовых документов в среде WWW.

**Тэг** – код (набор символов), идентифицирующий некоторый элемент документа и обозначающий способ отображения этого элемента.

**Браузер** - специальная программа, предназначенная для просмотра HTML-документов.

**Гипертекст -** текст, представленный в виде ассоциативно связанных автономных блоков.

**Гиперссылка -** средство указания смысловой связи фрагмента одного документа с другим документом или его фрагментом.

***Вводная часть***

Таблица, размещенная на Web-странице, может содержать в своих ячейках практически любую информацию: фрагмент текста, рисунок, комбинацию текста и рисунка (с заданным обтеканием), ссылки на другие страницы и пр., включая, в том числе, и вложенные другие таблицы. В HTML-документе таблице соответствует структура вложенных друг в друга контейнеров, схематически показанная ниже:

| | |
|---|---|
| <TABLE> | начало контейнера таблицы |
| <TR> | начало контейнера первой строки таблицы |
| <TD> | начало контейнера первой ячейки первой строки |

| | |
|---|---|
| … | содержимое этой ячейки |
| </TD> | конец контейнера первой ячейки первой строки |
| <TD> | начало контейнера второй ячейки первой строки |
| … | содержимое этой ячейки |
| </TD> | конец контейнера второй ячейки первой строки |
| <TD> | начало контейнера третьей ячейки первой строки |
| … | содержимое этой ячейки |
| </TD> | конец контейнера третьей ячейки первой строки |
| … | контейнеры других ячеек строки |
| </TR> | конец контейнера первой строки таблицы |
| <TR> | начало контейнера второй строки таблицы |
| … | далее аналогично |
| </TR> | конец контейнера последней строки таблицы |
| </TABLE> | конец контейнера таблицы |

Таким образом, здесь содержимое каждой ячейки таблицы, прежде всего, заключается в контейнер <TD>…</TD>. Затем все такие контейнеры, соответствующие ячейкам одной строки таблицы, заключаются в контейнер <TR>…</TR>, что кроме всего прочего указывает браузеру ширину таблицы (количество ячеек в ней по горизонтали). И наконец, все получившиеся контейнеры <TR>...</TR> заключаются в контейнер <TABLE>...</TABLE>, являющийся признаком таблицы и заодно указывающий браузеру высоту этой таблицы (количество строк в ней).

*Параметры тэга <TABLE>*

• BORDER — толщина разлиновки таблицы в пикселях или нуль, если разлиновку необходимо отключить (таблицы с «невидимой разлиновкой» — это наиболее удобный в HTML способ размещения текста и графики, а также многоколонной верстки); если параметр BORDER отсутствует, разлиновка также отключена.

• WIDTH — ширина таблицы (может указываться в пикселях или в процентах от ширины всей Web-страницы, тогда после числового значения, присваиваемого данному параметру, записывается знак «%»); если содержимое таблицы требует большего ее размера, чем это указано в параметре WIDTH, значение последнего игнорируется.

• CELLPADDING и CELLSPACING — отступ в пикселях содержимого ячеек от их границ (по умолчанию равен 1 пикселю).

• BGCOLOR — цвет фона таблицы.

*Параметры тэга <TR>*

• ALIGN — значение LEFT, CENTER или RIGHT указывает горизонтальное выравнивание содержимого для всех ячеек данной строки.

• VALIGN — определяет вертикальное выравнивание содержимого всех ячеек строки: значение TOP предписывает располагать все текстовые строки по верху ячеек, BOTTOM — по низу, CENTER — центрировать их по высоте ячейки.

• BGCOLOR — здесь этот параметр определяет отдельный цвет фона только для ячеек данной строки таблицы.

*Параметры тэга <TD>*

• WIDTH — ширина ячейки таблицы (в пикселях или в процентах относительно ширины всей таблицы, в последнем случае после числового значения записывается знак «%»); заметим, что важным является указание ширины ячеек только в первой строке таблицы, ширина всех последующих ячеек автоматически устанавливается по расположенным в первой строке, даже если для них установлены иные значения параметра WIDTH.

• ALIGN — здесь значение LEFT, CENTER или RIGHT определяет горизонтальное выравнивание содержимого только для данной ячейки.

• VALIGN — значение TOP, CENTER или BOTTOM определяет вертикальное выравнивание содержимого только для данной ячейки.

• BGCOLOR — определяет отдельный цвет фона для данной ячейки.

• BACKGROUND — позволяет задать для данной ячейки отдельное фоновое изображение.

• NOWRAP — отключение автопереноса текстовых строк в пределах ячейки.

• COLSPAN и ROWSPAN — объединение нескольких ячеек в одну по горизонтали и вертикали, соответственно.

Заметим, что одинаковые параметры, записанные в составе тэгов <TABLE>, <TR> и <TD>, подчиняются следующему правилу: приоритет параметра, распространяющегося на меньшую область, всегда выше.

С помощью тэга <TH> можно создать табличный заголовок. Текст элемента <TH> центрируется и выделяется жирным шрифтом.

***Пример 1***

```
<TABLE  BORDER=l>
<TR>
<TH>Башня</TH>
<TH>Год сооружения</TH>
<TH>Архитектор</TH>
<TH>Высота</TH>
</TR>
<TR>
<TD>Спасская</TD>
<TD ALIGN=CENTER">1491</TD>
<TD ALIGN=CENTER">П.Соларио</TD>
<TD ALIGN=CENTER">71</TD>
</TR>
…
<TD>Беклемишевская</TD>
<TD ALIGN=CENTER">1487</TD>
<TD ALIGN=CENTER>М.Руффо</TD>
<TD ALIGN=CENTER">46</TD>
</TR>
</TABLE>
```

| Башня | Год сооружения | Архитектор | Высота |
|---|---|---|---|
| Спасская | 1491 | П. Соларио | 71 |
| Никольская | 1491 | П. Соларио | 70 |
| Арсенальная | 1492 | П. Соларио | 60 |
| Троицкая | 1495 | П. Соларио | 80 |
| Боровицкая | 1490 | П. Соларио | 54 |
| Водовзводная | 1488 | А. Джилярди | 61 |
| Беклемишевская | 1487 | М. Руффо | 46 |

С помощью атрибута COLSPAN (ROWSPAN) можно указать, на сколько столбцов (строк) должна быть растянута указанная ячейка.

***Пример 2***

```
<TABLE BORDER=1>
<TR>
<TD>Ячейка 1</TD>
```

<TD>Ячейка 2</TD>
</TR>
<TR>
<TD COLSPAN=2> Ячейка 3, растянутая на 2 столбца</TD>
</TR>
</TABLE>

| Ячейка 1 | Ячейка 2 |
|---|---|
| Ячейка 3,растянутая на 2 столбца | |

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Создать HTML-документ, результат работы которого представлен на рисунке:

| Страна | Население (млн ) |
|---|---|
| Россия | 141 |
| США | 309 |
| Китай | 1338 |
| Великобритания | 61 |

***Задание 2.*** Опишите результат работы фрагмента HTML-документа:

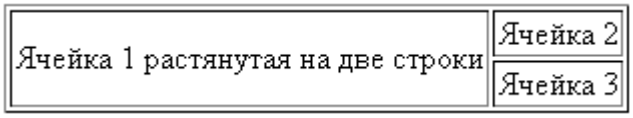
```
<table border='1'>
<tr>
<th>Заголовок 1</th>
<th>Заголовок 2</th>
<th>Заголовок 3</th>
</tr>
<tr>
<td>Ячейка 1</td>
<td>Ячейка 2</td>
<td>Ячейка 3</td>
</tr>
<tr>
<td>Ячейка 4</td>
<td>Ячейка 5</td>
<td>Ячейка 6</td>
</tr>
</table>
```

***Задание 3.*** Опишите результат работы HTML-документа:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Объединение ячеек</title>
</head>
```

```
<body>
<table border="1" cellpadding="4" cellspacing="0">
<tr>
<td rowspan="2">Браузер</td>
<th colspan="2">Internet  Explorer</th>
<th colspan="3">Opera</th>
<th colspan="2">Firefox</th>
</tr>
<tr>
<th>6.0</th><th>7.0</th><th>7.0</th><th>8.0</th><th>9.0</th><th>1.0</th><th>2.0</th>
</tr>
<tr align="center">
<td>Поддерживается</td>
<td>Нет</td><td>Да</td><td>Нет</td><td>Да</td><td>Да</td><td>Да</td><td>Да</td>
</tr>
</table>
</body>
</html>
```

***Задание 4.*** Создать HTML-документ, результат работы которого представлен на рисунке:




**ПРАКТИЧЕСКАЯ РАБОТА № 4. использование интерактивных форм на WEB-страницах**

***Тема:*** использование интерактивных форм на Web-страницах.

***Цель занятия:*** сформировать у обучаемых навыки использования интерактивных форм на Web-страницах.

***Основные понятия***

**Web-сайт** — совокупность Web-страниц с повторяющимся дизайном (не обязательно), объединенных по смыслу, навигационно и физически находящихся на одном Web-сервере.

**Web-сервер** — сервер, хранящий и предоставляющий во внешнюю сеть данные, организованные в виде Web–страниц.

**Язык гипертекстовой разметки HTML (HyperText Markup Language)** — стандартный язык, предназначенный для создания гипертекстовых документов в среде WWW.

**Тэг** – код (набор символов), идентифицирующий некоторый элемент документа и обозначающий способ отображения этого элемента.

**Браузер** - специальная программа, предназначенная для просмотра HTML-документов.

**Гипертекст -** текст, представленный в виде ассоциативно связанных автономных блоков.

**Гиперссылка -** средство указания смысловой связи фрагмента одного документа с другим документом или его фрагментом.

**HTML-форма -** отдельная Web-страница или часть страницы, на которой размещены разнообразные типовые для Windows элементы интерактивного диалога: поля ввода текста, флажки и радиокнопки с поясняющими строками текста, раскрывающиеся списки и одна или несколько кнопок, обычно служащих для

очистки формы (приведения в исходный вид в случае неправильного заполнения) и отправки введенных данных (сформированного информационного запроса) на сервер.

***Вводная часть***

Интерактивные формы предназначены для сбора информации от посетителей Web-страницы. Внешне форма выглядит, как анкета, в которую посетитель может вписать свой текст или выбрать значения из предлагаемых перечней.

*Формы.* Форма помещается в специальный контейнер <FORM>…</FORM>, внутри которого располагаются все тэги элементов формы. Там же обязательно находится кнопка, отправляющая заполненную анкету на сервер для последующей обработки данных.

Поля формы создаются с помощью тэгов <INPUT>, <SELECT> и <TEXTAREA>, в которых с помощью атрибутов задаются параметры полей. С помощью атрибута TYPE задается тип поля, атрибут NAME присваивает полю имя, а атрибут VALUE содержит его значение. Для различных конкретных типов поля существуют также дополнительные атрибуты.

При обработке формы на сервер передаются пары NAME=VALUE, где VALUE — значение соответствующего атрибута, установленное пользователем или заданное по умолчанию.

*Текстовые поля.* Текстовое поле создается тэгом:

```
<INPUT TYPE="text" NAME="regist" VALUE=" ">
```

Атрибут VALUE принимает такое значение, которое задается пользователем при заполнении поля. Если пользователь ввел в поле текст «Фамилия», то на сервер будет передано regist=Фамилия.

*Флажки.* Флажки могут объединяться в группы присвоением атрибутам NAME всех флажков одинакового значения. Для установки флажка по умолчанию используется атрибут CHECKED. Группа флажков создается тэгами:

```
<INPUT TYPE="checkbox" NAME="chb1" VALUE=1 CHECKED>
<INPUT TYPE="checkbox" NAME="chb1" VALUE=2>
```

На сервер передается значение атрибута VALUE флажка, установленного пользователем. Если пользователем установлены несколько флажков, то на сервер будут переданы значения уставленных флажков группы через запятую chb1=1,2.

*Переключатели.* Переключатели существуют только в составе группы, что обеспечивается присвоением атрибутам NAME всех переключателей одинакового значения. Для установки переключателя по умолчанию используется атрибут CHECKED. Группа переключателей создается тэгами:

```
<INPUT TYPE="radio" NAME="rad1" VALUE=1 CHECKED>
<INPUT TYPE="radio" NAME="rad1" VALUE=2>
```

На сервер передается значение атрибута VALUE переключателя, установленного пользователем. Если пользователем установлен первый переключатель, то на сервер будет передано rad1=1.

*Списки.* Списки предоставляют пользователю выбор элементов в форме ниспадающего меню (значение атрибута SIZE=1) или списка прокрутки. Список помещается в контейнер:

```
<SELECT NAME="list" SIZE=N>
<OPTION SELECTED>Первый</OPTION>
<OPTION>Второй</OPTION>
<OPTION>Третий</OPTION>
</SELECT>
```

На сервер передается значение атрибута OPTION, выбранного пользователем или установленного по умолчанию атрибутом SELECTED. Если пользователем выбран третий элемент списка, то на сервер будет передано list=Третий.

*Текстовая область.* Текстовая область представляет собой текстовое поле с заданным количеством строк (значение атрибута ROWS) и столбцов (значение атрибута COLS). Создается тэгом:

<TEXTAREA NAME="resume" ROWS=M COLS=N>

Текст по умолчанию </TEXTAREA>.

На сервер передается содержимое поля, если пользователь не изменял текст, то resume=Текст по умолчанию.

*Кнопки.* На форме должны присутствовать кнопки, которые реализуют отправку данных из формы для обработки на сервер и очистку формы от введенных данных.

Кнопка отправки данных формы реализуется с помощью тэга:

<INPUT TYPE="submit" VALUE="Отправить">

Кнопка очистки данных формы реализуется с помощью тэга:

<INPUT TYPE="reset" VALUE="Очистить">

***Описание работы***

Выполнить задания. Задания выполняются каждым обучающимся в отдельности.

***Задание 1.*** Опишите результат работы фрагмента HTML-документа:

```
<form>
<p> Введите ФИО в поля ниже: </p> <br />
Имя: <input type="text" name="firstname" /><br />
Фамилия: <input type="text" name="lastname" /><br />
Отчество: <input type="text" name="lastname" />
</form>
```

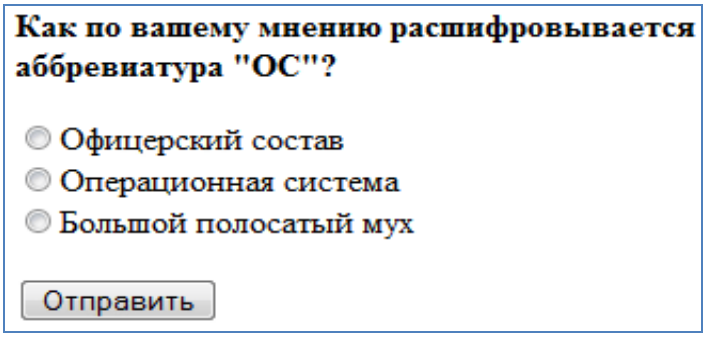
***Задание 2.*** Опишите результат работы фрагмента HTML-документа:

```
<form>
<p> Укажите Ваш пол: </p>
<input type="radio" name="s" value="m" /> Мужской<br />
<input type="radio" name="s" value="f" /> Женский
</form>
```

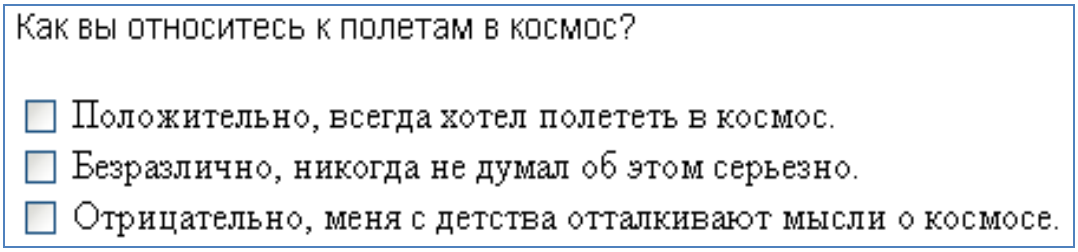
***Задание 3.*** Опишите результат работы фрагмента HTML-документа:

```
<p> Выберите ваш пол </p>
<form>
<select name="sex" >
<option value="m"> Мужской </option>
<option value="f"> Женский </option>
</select>
</form>
```

***Задание 4.*** Создать HTML-документ, результат работы которого представлен на рисунке:




***Задание 5.*** Создать HTML-документ, результат работы которого представлен на рисунке:




**ЛИТЕРАТУРА**

**Основная учебная**

**Основная учебная**

1. Белянина Н.В., Корнеева Е.В. Продвижение сайтов и их регистрация в поисковых системах [Электронный ресурс]: рабочий учебник/ Белянина Н.В., Корнеева Е.В. - 2014. - http://lib.muh.ru
2. Белянина Н.В., Корнеева Е.В. Язык гипертекстовой разметки HTML [Электронный ресурс]: рабочий учебник/ Белянина Н.В., Корнеева Е.В. - 2014. - http://lib.muh.ru
3. Сычев А.В. Перспективные технологии и языки веб-разработки [Электронный ресурс]/ Сычев А.В.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 493 c.— http://www.iprbookshop.ru/39643.— ЭБС «IPRbooks»

**Дополнительная**

1. Бугреев В.А. Информационные сервисы INTERNET. Администрирование серверов World Wide Web [Электронный ресурс]: рабочий учебник/ Бугреев В.А. - 2009. - http://lib.muh.ru
2. Бугреев В.А. Администрирования сетей TCP/IP и информационных технологий INTERNET. Основы межсетевого обмена в сетях TCP/IP [Электронный ресурс]: рабочий учебник/ Бугреев В.А. - 2009. - http://lib.muh.ru
3. Корнеева Е.В., Принципы построения Web-серверов. Основные принципы и технологии их организации и функционирования [Электронный ресурс]: рабочий учебник/ Корнеева Е.В. - 2010. - http://lib.muh.ru

***Материально-техническое обеспечение:***

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**
ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ
ПО ДИСЦИПЛИНЕ «ПРИНЦИПЫ ПОСТРОЕНИЯ WEB-СЕРВЕРОВ»
(для направления подготовки 09.03.01 «Информатика
и вычислительная техника»)

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки Е.В. Белюсенко

**4924.01.01;МУ.01;1**

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ) ПО ДИСЦИПЛИНЕ «ЭЛЕКТРОТЕХНИКА, ЭЛЕКТРОНИКА И СХЕМОТЕХНИКА»

**МОСКВА 2018**

Разработано Д.П. Гуриным

Под ред. С.Е. Федорова, к.т.н., проф.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ) ПО ДИСЦИПЛИНЕ «ЭЛЕКТРОТЕХНИКА, ЭЛЕКТРОНИКА И СХЕМОТЕХНИКА»

Учебный продукт по дисциплине «Электротехника, электроника и схемотехника» разработан в соответствии с учебным планом по направлению подготовки «Информатика и вычислительная техника» на основании требований Федерального государственного образовательного стандарта высшего образования (ФГОС ВО) к минимуму содержания и уровню подготовки бакалавра по направлению 09.03.01 «Информатика и вычислительная техника».

Методические указания подготовлены для обучающихся образовательной организации и предназначены для практического освоения компьютерных технологий моделирования электрических и электронных цепей, реальных исследований электрических цепей на универсальном лабораторном стенде "Миниатюрная электротехническая лаборатория МЭЛ" под общим наименованием «Профессиональные лабораторные занятия» (ПЛЗ) по направлению 09.03.01 «Информатика и ВТ» в рамках дисциплины «Электротехника, электроника и схемотехника».

**Цель ПЛЗ** заключается в практическом освоении обучающимися экспериментальных методов исследования электрических цепей и электронных схем, компьютерных технологий моделирования электрических и электронных цепей, закреплении теоретических знаний и навыков расчета электрических цепей и электронных схем, знакомстве с электрическими измерениями.

***Особенность*** данного вида занятий заключается в последовательности осуществления практических и познавательных действий.

В результате выполнения лабораторных работ обучающийся должен:

***знать:***

- ✓ фундаментальные физические законы и соотношения в области электричества и магнетизма, методы расчета и основные свойства электрических цепей, способы преобразования линейных электрических схем;
- ✓ фундаментальные понятия, положения и принципы в области электроники;
- ✓ основные технические параметры и характеристики электрических и электронных устройств;
- ✓ принципы построения, параметры и характеристики цифровых и аналоговых элементов ЭВМ;
- ✓ современные подходы к анализу и синтезу электронных устройств и современные технологии их создания;

***уметь:***

- ✓ проводить исследования электрических и электронных устройств;
- ✓ анализировать прохождение сигналов через различные электронные устройства;
- ✓ ставить и решать схемотехнические задачи, связанные с выбором системы элементов при заданных требованиях к параметрам;
- ✓ тестировать, испытывать и использовать программно-аппаратные средства вычислительных и информационных систем;
- ✓ проводить компьютерный анализ цифровых устройств.

В процессе выполнения лабораторных работ у обучающегося формируются навыки:

- ✓ опытного исследования свойств линейной электрической цепи;
- ✓ нахождения токов в ветвях электрической цепи;
- ✓ определения потенциалов точек электрической цепи;
- ✓ исследования амплитудных и фазовых соотношений в цепях переменного тока, частотных характеристик и резонансных явлений;
- ✓ построения векторных диаграмм токов и напряжений;
- ✓ определения параметров магнитно-связанных катушек;
- ✓ исследования амплитудно-частотных характеристик простого параллельного контура и сложных контуров;
- ✓ определения резонансной частоты, полосы пропускания, добротности параллельного контура и сложных контуров;
- ✓ определение параметров линейного пассивного четырехполюсника по входным сопротивлениям в режимах холостого хода и короткого замыкания;
- ✓ исследования переходных процессов в цепях первого порядка R,L и R,C, а также в цепях второго порядка R,L,C при апериодическом и колебательном характерах процесса;
- ✓ исследования цепей с периодическими несинусоидальными токами;

✓ исследования режимов работы трехфазных цепей переменного тока при различных способах соединения симметричных и несимметричных нагрузок;

✓ исследования динамических характеристик асинхронного двигателя;

✓ исследования вольтамперных характеристик схем на основе биполярного транзистора;

✓ исследования вольтамперных характеристик схем на основе полевого транзистора;

✓ исследования операционных усилителей в цепях постоянного и переменного токов;

✓ исследования характеристик схем с цифро-аналоговыми преобразователями;

✓ исследования и синтеза логических схем;

✓ исследования характеристик триггеров.

Каждое занятие содержит теоретические сведения, практические задания, методику их выполнения.

***Каждое занятие подразделяется на следующие части:***

***Первая – вступительная.*** Обучаемые знакомятся с темой и целью занятия, перечнем приборов или прикладного программного обеспечения для проведения исследования или моделирования электрических цепей и электронных схем.

***Вторая – теоретическая.*** Обучаемые самостоятельно изучают теоретические сведения по теме занятия.

***Третья – практическая.*** Обучаемые самостоятельно выполняют практические задания по теме занятия в соответствии с методикой их выполнения, работая на стенде «МЭЛ» или используя ПО MicroCap.

***Четвёртая – заключительная.***

Предназначена для подведения итогов, контроля качества усвоения материала. Подводятся итоги занятия, обучаемым выставляются оценки.

Для проведения лабораторных работ на универсальном лабораторном стенде МЭЛ студенческая группа делится на подгруппы по 2–3 человека. Состав групп сохраняется до конца семестра. Обработка результатов опытов проводится каждым обучающимся самостоятельно.

Лабораторные работы с использованием ПО MicroCap выполнятся каждым обучающимся индивидуально.

**Перечень профессиональных лабораторных занятий**

| | |
|---|---|
| ***Лабораторное занятие № 1*** Исследование линейной электрической цепи постоянного тока (универсальный лабораторный стенд МЭЛ) | ✓ опытное исследование свойств линейной электрической цепи;<br>✓ нахождение токов в ветвях методом наложения и по законам Кирхгофа;<br>✓ определение потенциалов точек электрической цепи;<br>✓ исследование передачи энергии от активного двухполюсника нагрузке;<br>✓ сопоставления опытных и теоретических данных |
| ***Лабораторное занятие № 2*** Исследование цепей переменного тока (универсальный лабораторный стенд МЭЛ) | ✓ исследование амплитудных и фазовых соотношений в цепях переменного тока;<br>✓ исследование частотных характеристик и резонансных явлений;<br>✓ построение векторных диаграмм токов и напряжений |
| ***Лабораторное занятие № 3*** Исследование последовательного колебательного контура (универсальный лабораторный стенд МЭЛ) | ✓ исследование амплитудно-частотных характеристик последовательного колебательного контура;<br>✓ определение резонансной частоты $f_р$;<br>✓ полосы пропускания П;<br>✓ добротности $Q$ контуров |
| ***Лабораторное занятие № 4*** Трехфазные электрические цепи (универсальный лабораторный стенд МЭЛ) | ✓ исследование режимов работы трехфазных цепей переменного тока при различных способах соединения симметричных и несимметричных нагрузок;<br>✓ построение векторных диаграмм токов и напряжений в трехфазных цепях |
| ***Лабораторное занятие № 5*** | ✓ определение параметров линейного пассивного четырехполюсника по |

| | |
|---|---|
| Исследование четырехполюсника (универсальный лабораторный стенд МЭЛ) | входным сопротивлениям в режимах холостого хода и короткого замыкания;<br>✓ выбор сопротивления нагрузки четырехполюсника из условия выделения в ней максимальной активной мощности |
| ***Лабораторное занятие № 6***<br>Исследование переходных процессов в цепях с сосредоточенными параметрами (универсальный лабораторный стенд МЭЛ) | ✓ исследование переходных процессов в цепях первого порядка R,L и R,C;<br>✓ исследование переходных процессов в цепи второго порядка R,L,C при апериодическом и колебательном характерах процесса |
| ***Лабораторное занятие № 7***<br>Электрические цепи с магнитно-связанными катушками (универсальный лабораторный стенд МЭЛ) | ✓ определение параметров магнитно-связанных катушек;<br>✓ изучение распределения токов, напряжений и мощностей в цепях с взаимной индуктивностью |
| ***Лабораторное занятие № 8***<br>Исследование характеристик биполярного транзистора и усилителя на биполярном транзисторе (универсальный лабораторный стенд МЭЛ) | ✓ исследование вольтамперных характеристик биполярного транзистора;<br>✓ исследование усилителя на основе биполярного транзистора |
| ***Лабораторное занятие № 9***<br>Исследование характеристик полевого транзистора и усилителя на полевом транзисторе (универсальный лабораторный стенд МЭЛ) | ✓ исследование вольтамперных характеристик полевого транзистора;<br>✓ исследование усилителей на основе полевого транзистора |
| ***Лабораторное занятие № 10***<br>Исследование операционных усилителей в цепях постоянного и переменного токов (универсальный лабораторный стенд МЭЛ) | ✓ изучение типовых функциональных схем включения операционных усилителей;<br>✓ исследование свойств схем включения операционных усилителей |

**Материально-техническое и информационное обеспечение учебного занятия**

**а) Материально-техническое обеспечение:**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**б) Информационное обеспечение.**

– Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы.
- тренинговые и тестирующие программы.
- интеллектуальные роботизированные системы оценки качества выполненных работ.

– Информационные и роботизированные системы, программные комплексы, программное обеспечение для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ПО «Комбат»;
- ПО «ЛиК»;
- ПК «КОП»;
- ИР «Каскад».

**Литература**

**Основная учебная**

1. **Лаппи Ф.Э.** Минимальный курс электротехники и электроники. Часть 1. Основные элементы электротехники и электроники [Электронный ресурс]: учебное пособие/ Лаппи Ф.Э.— Электрон. текстовые данные.— Новосибирск: Новосибирский государственный технический университет, 2014.— 112 с.— http://www.iprbookshop.ru/45112.— ЭБС «IPRbooks»

2. **Трубникова В.Н.** Электротехника и электроника. Часть 1. Электрические цепи [Электронный ресурс]: учебное пособие/ Трубникова В.Н.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2014.— 137 с. http://www.iprbookshop.ru/33672.— ЭБС «IPRbooks»

3. **Букштынович, И.М.** Микропроцессорные БИС/СБИС. Интерфейсные БИС/СБИС в микропроцессорных комплектах. [Электронный ресурс]: рабочий учебник/ Букштынович, И.М. - 2013. - http://lib.muh.ru.

4.. **Гурин, Д.П.** Электронные устройства и преобразователи. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

5. **Гурин, Д.П.** Электронные приборы. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

6. **Гурин, Д.П.** Трансформаторы, электрические машины, электроизмерительные приборы и электрические измерения. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

**Дополнительная**

1. **Дураджи, В.Н.** Электрические цепи при несинусоидальных токах и напряжениях. Магнитные цепи. [Электронный ресурс]: рабочий учебник/ Дураджи, В.Н. - 2011. - http://lib.muh.ru.

2. **Гурин, Д.П.** Четырехполюсники. Электрические фильтры. Переходные процессы в линейных электрических цепях. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2011. - http://lib.muh.ru.

3. **Гурин, Д.П.** Электрические цепи при постоянных и синусоидальных токах и напряжениях. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2011. - http://lib.muh.ru.

## Лабораторная работа № 1. Исследование линейной электрической цепи постоянного тока

***Продолжительность: 90 минут.***

***Дисциплина «Электротехника, электроника и схемотехника». Юнита 1.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

### 1 Инструкция по технике безопасности

#### *1.1 Общие правила безопасности при работе обучающихся в учебной лаборатории*

В помещениях лаборатории категорически запрещается трогать, включать и выключать без разрешения преподавателя или дежурного сотрудника лаборатории любую аппаратуру.

При работе в лаборатории выполняйте только ту работу, которая Вам поручена.

Категорически запрещается производить другую работу.

Во время выполнения заданий не ходите без дела по лаборатории, так как этим Вы отвлекаете внимание товарищей и оставляете без наблюдения свою установку, что может повлечь за собой несчастный случай.

Немедленно сообщайте преподавателю или дежурному сотруднику лаборатории о замеченных Вами неисправностях и нарушениях правил техники безопасности.

Если с Вами или с Вашим товарищем произошел несчастный случай, немедленно сообщите об этом преподавателю или сотруднику лаборатории.

По окончании работы выключите установку и приведите в порядок свое рабочее место. После уборки сообщите сотруднику лаборатории об окончании работы и только после этого Вы можете оставить лабораторию.

Прежде чем приступить к выполнению лабораторной работы**,** необходимо изучить инструкцию по технике безопасности к этой работе.

### *1.2 Правила электробезопасности при работе в лаборатории*

***Особенности действия тока на живую ткань***

Действие электрического тока на живую ткань в отличие от действия других материальных факторов носит своеобразный и разносторонний характер. Так, электрический ток, проходя через живой организм, производит термическое и электролитическое действия, являющиеся обычными физико-химическими процессами, присущими как живой, так и неживой материи. Вместе с тем электрический ток производит и биологическое действие, которое является особым специфическим процессом, свойственным лишь живой ткани.

Термическое действие тока проявляется при ожогах тела, нагреве и повреждении кровеносных сосудов, перегреве нервов, сердца, мозга и других органов, что вызывает в них серьезные функциональные расстройства.

Электролитическое действие тока проявляется в разложении органической жидкости, в том числе крови, вызывая, тем самым, значительные нарушения их физико-химических составов, а также в ткани в целом.

Биологическое действие тока выражается главным образом в нарушении биологических процессов, протекающих в нормально действующем организме и теснейшим образом связанных с его жизненными функциями.

Поражение людей электрическим током может произойти как при высоком, так и при низком напряжении, причем поражение током даже при низком напряжении в ряде случаев приводит к смертельному исходу.

Однако многие исполнители работ при обслуживании электроустановок низкого напряжения пренебрегают требованиями техники безопасности, ошибочно считая опасным для жизни только высокое напряжение.

Большое значение в случае поражения человека электрическим током имеют окружающие условия. В зависимости от окружающих условий изменяется и величина опасного для человека напряжения. По степени опасности поражения людей электрическим током рабочие помещения разделяются на три группы:

а) помещения без повышенной опасности;

б) помещения с повышенной опасностью;

в) помещения особо опасные.

Помещения с повышенной опасностью характеризуются наличием одного из следующих условий, создающих повышенную опасность:

- ✓ токопроводящей пыли;
- ✓ влажности или сырости (относительная влажность воздуха превышает 75 %);
- ✓ токопроводящих полов (металлических, земляных, кирпичных и т.п.);
- ✓ возможности одновременного прикосновения человека к металлоконструкциям здания, технологическим аппаратам, механизмам, имеющим соединения с землей, с одной стороны, и к металлическим корпусам электрооборудования – с другой (коэффициент заполнения помещения электрооборудованием превышает 0,2);
- ✓ высокой температуры (выше +30 °С).

Особо опасные помещения характеризуются наличием одного из следующих условий, создающих особую опасность:

- ✓ особой сырости (относительная влажность воздуха близка к 100 %; потолок, стены, пол и предметы, находящиеся в помещении, покрыты влагой);
- ✓ химически активной среды, где по условиям работы постоянно или длительно содержатся пары или образуются отложения, действующие разрушающе на электрическую изоляцию и токоведущие части электрооборудования;
- ✓ одновременно двух или более условий повышенной опасности.

К помещениям без повышенной опасности относятся помещения, в которых отсутствуют условия, создающие повышенную опасность или особую опасность, т.е.:

✓ сухие (относительная влажность воздуха не выше 75 %);

✓ отапливаемые (температура воздуха от + 5 до +30 °С);

✓ помещения, в которых коэффициент заполнения оборудованием не превышает 0,2 его площади;

✓ нет токопроводящих полов, токопроводящей пыли, химически активной среды.

Для различных помещений в зависимости от их групп принято напряжение, превышение которого считается опасным для жизни человека:

а) в помещениях без повышенной опасности и с повышенной опасностью – 36 В;

б) в помещениях особо опасных – 12 В.

Во всех случаях должно быть обеспечено правильное выполнение защитного заземления корпусов электрооборудования и приборов. Расположение рабочих мест должно быть таково, чтобы исключалась возможность одновременного прикосновения к корпусам электрооборудования и заземленным конструкциям.

В электрических установках возможны случаи, когда металлические конструктивные части, нормально не находящиеся под напряжением, получают по различным причинам потенциал, отличный от потенциала «земли». Прикосновение к частям оборудования под таким потенциалом вызывает прохождение через тело человека тока, могущего представить опасность для его жизни. Поэтому для обеспечения безопасности людей, работающих с электрическими установками, требуется выполнять защитное заземление или зануление.

Защитным заземлением называется соединение с заземлителем металлических, изолированных от напряжения частей электроустановок. Выполняется для защиты от опасных напряжений прикосновения. При повреждении изоляции оборудования или замыкания сети на корпус заземленного оборудования ток проходит через заземление в землю.

Это обеспечивает снижение напряжения прикосновения до безопасной величины. Защитное заземление применяется в сетях с напряжением до 1000 В, работающих с изолированной нейтралью, и в сетях с напряжением выше 1000 В как с изолированной, так и с заземленной нейтралью.

Защитным занулением называется намеренное соединение металлических токопроводящих частей установки, которые могут случайно оказаться под напряжением, с «нулевым» (заземленным) проводом.

Зануление служит для быстрого отключения установки. Защитное зануление применяется в сетях с напряжением до 1000 В, работающих с заземленной нейтралью.

Применение в одной и той же сети зануления для одних частей электрооборудования и заземления для других не допускается**.**

Электроустановки необходимо заземлять или занулять:

✓ при напряжении 500 В и выше;

✓ при напряжении выше 36 В в помещениях с повышенной опасностью, особо опасных и в наружных электроустановках; при всех напряжениях переменного и постоянного тока;

✓ во взрывоопасных помещениях.

Заземление электроустановок не требуется при номинальных значениях напряжения 36 В и ниже во всех случаях, за исключением взрывоопасных установок.

Заземлению или занулению подлежат: металлические корпуса электрических машин, трансформаторов, светильников, каркасы распределительных щитков, щитков или шкафов управления, металлические корпуса кабельных муфт, металлические оболочки и металлические защитные трубы проводов, кабелей и т.п.

Заземлению или занулению не подлежат: арматура подвесных и штыри опорных изоляторов; оборудование, установленное на заземленных металлических конструкциях; корпуса измерительных приборов, реле и т.п.

Каждый заземленный элемент установки должен быть присоединен к заземлителю или заземляющей магистрали посредством отдельного ответвления.

Последовательное включение в заземляющий проводник нескольких заземляемых частей установки запрещается**.**

Присоединение заземляющих проводников к заземлителям и заземляющим конструкциям должно быть выполнено сваркой, а к корпусам аппаратов, машин и т.п. – сваркой или надежным болтовым соединением, при этом в сырых помещениях с едкими парами или газами контактные поверхности должны иметь защитные покрытия. Концы заземляющих гибких проводников, применяемых для присоединения к корпусам приборов, аппаратов и т.д., должны иметь приварные наконечники.

Заземление оборудования, подвергающегося частому монтажу или установленного на движущихся частях машин, должно выполняться при помощи гибких проводников приварными к ним наконечниками.

Переносное заземление является обязательной мерой защиты работающего от случайного появления напряжения на месте работы, а также от поражения зарядом высоковольтных конденсаторов. Для переносного заземления должен применяться медный многожильный голый провод. Применение для переносного заземления провода в изоляции запрещается. Запрещается использовать в качестве заземляющих проводников металлические оболочки проводов, металлизированные изоляционные трубки. Открыто проложенные голые проводники и голые сети заземления должны быть окрашены в черный цвет. Допускается окраска открытых заземляющих проводников в другие цвета в соответствии с оформлением помещения, но при этом они должны иметь в местах соединения и ответвления не менее чем две полосы черного цвета на расстоянии 150 мм друг от друга.

Заземляющие проводники, расположенные в помещениях, должны быть доступны для осмотра.

Надежность заземления и его общее состояние должны проверяться путем замеров не реже одного раза в год, а также после каждого капитального ремонта и длительного бездействия установки. Результаты проверок заземления должны записываться в журнале. После аварии заземляющие проводники (шины) и места контактов (соединений и присоединений) должны проверяться внешним осмотром. Внешний осмотр состояния заземляющих проводников (шин) должен производиться не реже 1 раза в 6 месяцев, а в сырых и особо опасных помещениях – не реже одного раза в 3 месяца.

При нарушении исправности заземления установка должна быть немедленно отключена до устранения этой неисправности**.**

Для определения технического состояния заземляющего устройства должны периодически производиться:

✓ внешний осмотр видимой части заземляющего устройства;

✓ осмотр с проверкой наличия цепи между заземлителем и заземляемыми элементами (отсутствие обрывов и неудовлетворительных контактов в проводке, соединяющей прибор с заземляющим устройством), а также проверка пробивных предохранителей трансформаторов; измерение сопротивления заземляющего устройства;

✓ измерение полного сопротивления петли «фаза-нуль»;

✓ проверка надежности соединений естественных заземлителей.

Наибольшие допустимые сопротивления заземления составляют:

✓ для установки до 1000 В – не более 4 Ом;

✓ для установки выше 1000 В – не более 0,5 Ом.

Сечение заземляющих проводников определяется мощностью электроустановок.

Минимальное сечение заземляющих (зануляющих) проводников в электроустановках до 1000 В следующее: медных – 4,0 мм$^2$, алюминиевых – 6,0 мм$^2$.

На импульсных генераторах и на других установках, где, несмотря на большие напряжения, сила тока незначительна или очень мала, сечение переносного заземления берется из условий его механической прочности.

При ремонтных и монтажных работах на установках высокого напряжения после проверки отсутствия напряжения или в случае освобождения отключенных частей установки от остаточного заряда (конденсаторы,

емкость линий) на отключенные токоведущие части накладывается заземление. При этом переносное заземление должно быть подключено к «земле», т.е. к контуру заземления.

Тщательно следите за исправностью изоляции проводов и оборудования.

Немедленно сообщайте о замеченных неисправностях.

В процессе работы не отключайте и не обрывайте проводов защитного заземления.

***Строго воспрещается:***

- ✓ включать схему под напряжением без предварительной проверки и разрешения преподавателя;
- ✓ выключать силовые и осветительные рубильники без разрешения преподавателя;
- ✓ производить переключения в схемах, находящихся под напряжением;
- ✓ оставлять без наблюдения схему, находящуюся под напряжением;
- ✓ закорачивать блокирующие устройства;
- ✓ заходить за ограждения;
- ✓ протягивать руки за ограждения;
- ✓ работать с незаземленным электрооборудованием;
- ✓ снимать и перевешивать предупреждающие и запрещающие плакаты;
- ✓ работать одному в помещениях на установках с электрооборудованием.

***Классификация установок по напряжению***

Электроустановки классифицируются по напряжению.

Различают электроустановки до 1000 В и электроустановки выше 1000 В. Различают также электроустановки с большими токами замыкания на землю, в которых ток однополюсного глухого замыкания на землю превышает 500 А, и электроустановки с малыми токами замыкания на землю, в которых ток однополюсного глухого замыкания на землю равен или менее 500 А. Поэтому комплекс защитных мер должен соответствовать виду электроустановок и соответствовать условиям применения электрооборудования, обеспечивать достаточную безопасность.

Весьма существенно влияние на безопасность условий среды, от которых зависит состояние изоляции, а также электрическое сопротивление тела человека. Повышенная влажность снижает сопротивление изоляции. Кроме того, отмечено увеличение емкости гибких кабелей с резиновой изоляцией при повышенной влажности воздуха, что можно объяснить изменением диэлектрической проницаемости изоляции при изменении влажности.

Высоковольтные установки, применяемые в учебном процессе, при научно-исследовательских работах и при дипломных работах, представляют повышенную опасность электротравматизма. Поэтому все лица, работающие и обучающиеся в лаборатории, обязаны твердо знать и строго соблюдать требования техники безопасности к установкам высокого напряжения, инструкции по технике безопасности, знать назначение и устройство защитных приспособлений, а также правила освобождения пострадавшего от действия электрического тока и оказания ему первой помощи.

При работе с высоковольтными установками недопустимо присутствие в помещении лаборатории посторонних или случайных лиц.

Обучающиеся**,** не сдавшие экзамены по технике безопасности и поэтому не имеющие права самостоятельно работать**,** к работе в лаборатории на установке не допускаются**.**

Работа на установках высокого напряжения должна проводиться не менее чем двумя лицами, из которых одно должно иметь квалификацию не ниже третьей группы, которая дает право самостоятельно работать на установках высокого напряжения. На это лицо возлагается ответственность за соблюдение всех правил техники безопасности. Один человек не имеет права выполнять работы на установках высокого напряжения.

Действующая высоковольтная установка должна обеспечивать безопасность выполнения работ во всех случаях. Для этой цели служат: ограждения, защитное заземление, заземляющие штанги, блокировки, сигнализация, два видимых разрыва (рубильника), щитки питания, предупреждающие и запрещающие плакаты.

В каждой высоковольтной лаборатории должна быть общая инструкция по технике безопасности при работе с высоковольтными установками, составленная с учетом специфики помещений и условий работы в данной лаборатории.

Кроме того, каждая высоковольтная установка должна иметь свою специально составленную инструкцию по технике безопасности.

Инструкция должна содержать:

а) порядок включения и выключения установки;

б) перечень запрещенных действий;

в) краткое перечисление защитных средств и норм (ограждение, заземление, блокировки и правила использования их);

г) действия в аварийных ситуациях.

Часто меняющаяся обстановка в лаборатории обязывает не полагаться на защитное устройство, а всякий раз, прежде чем приступить к работе, убедиться:

- ✓ в наличии инструкции по технике безопасности для данной установки;
- ✓ в исправности схемы блокировки, переносных заземлителей и постоянного заземления;
- ✓ в исправности ограждения и правильности его установки;
- ✓ в наличии предупреждающих плакатов;
- ✓ в отсутствии за ограждением людей.

Включая высокое напряжение, необходимо проверить, включилась ли световая и звуковая сигнализация, предупреждающая о включении высокого напряжения.

В процессе работы при включенной установке категорически воспрещается**:**

- ✓ заходить за ограждение;
- ✓ передвигать ограждение;
- ✓ протягивать руки за ограждение;
- ✓ закорачивать или выключать блокировочное устройство;
- ✓ снимать запрещающие и предупреждающие плакаты;
- ✓ оставлять установку, находящуюся под напряжением, без присмотра.

Работающие с высоким напряжением должны помнить, что отключенный рубильник и наличие блокировки не свидетельствуют еще об отсутствии напряжения на элементах установки. На конденсаторах остается заряд; разряд конденсатора может нанести травму, даже смертельную. Поэтому конденсаторы должны быть разряжены, а затем заземлены. Напряжение должно быть снято, если возникают сомнения в исправности установки или защитных средств.

При временном прекращении работы, переключениях в схеме и ремонтных работах установка должна быть отключена от источника питания и на рубильнике вывешен плакат "НЕ ВКЛЮЧАТЬ", "РАБОТАЮТ ЛЮДИ".

После окончания работы необходимо:

- ✓ заземлить части установки, бывшие или могущие быть под высоким напряжением;
- ✓ снять предупреждающие и запрещающие плакаты;
- ✓ принять меры к предупреждению возможности случайного (ошибочного) включения установки под напряжение.

### *1.3 Правила помощи пострадавшему от электрического тока при несчастном случае*

Первая помощь пострадавшему от электрического тока состоит из двух этапов: освобождение пострадавшего от действия тока и оказание ему медицинской помощи.

Прикосновение к токоведущим частям электрических установок, находящихся под напряжением, вызывает в большинстве случаев судорожное сокращение мышц. Вследствие этого пальцы, в случае если пострадавший держит провод в руках, так сильно сжимаются, что самостоятельно выпустить провод из рук становится невозможным. Если пострадавший остается в соприкосновении с токоведущими частями, то следует, прежде всего, быстро освободить его от действия электрического тока. При этом необходимо помнить, что без принятия надлежащих мер предосторожности прикасаться к человеку, находящемуся под током, опасно для жизни. Первым действием должно быть быстрое отключение той части установки, которой касается пострадавший.

При этом необходимо учитывать следующее:

✓ в случае нахождения пострадавшего на высоте отключение установок и освобождение пострадавшего от тока могут вызвать падение его с высоты, в этом случае должны быть приняты меры, обеспечивающие безопасность падения пострадавшего, иначе неосмотрительное отключение может принести не меньший вред, чем электрический ток;

✓ при отключении установок может быть одновременно отключено электрическое освещение и надо позаботиться о других источниках света (фонарь, аварийное освещение, аккумуляторные фонари и т.д.), не задерживая при этом отключения установки и оказания помощи пострадавшему.

Если отключение установки не может быть произведено достаточно быстро, то необходимо принять меры к отделению пострадавшего от токоведущих частей.

***При низком напряжении***

Для отделения пострадавшего от токоведущих частей или провода следует воспользоваться сухой одеждой, сухим канатом, сухой палкой, доской или каким-либо другим сухим непроводником. Нельзя пользоваться в таких случаях металлическими или мокрыми предметами. Чтобы оторвать пострадавшего от токоведущих частей, можно взяться также за его одежду, если она суха и отстает от тела, например за полы, избегая при этом прикосновения к окружающим предметам и частям тела, не покрытым одеждой. Не следует также оттаскивать пострадавшего за ноги без предварительной хорошей изоляции своих рук, так как обувь может быть сырой, а находящиеся в ней гвозди или крючки для шнурков являются проводниками тока. Для изоляции рук при спасении, в особенности если необходимо коснуться частей тела пострадавшего, не покрытых одеждой, надо надеть резиновые перчатки или обмотать себе руки шарфом, опустить на руки свой рукав и т.п. или накинуть на пострадавшего резину, прорезиненную материю (плащ) или же просто сухую материю; можно также изолировать себя от земли, встав на сухую доску или какую-либо сухую, не проводящую ток подстилку, сверток одежды и т.д. При низком напряжении, когда ток проходит в землю через человека и последний судорожно сжимает в руках один провод, проще прервать ток, отделив пострадавшего от земли (например, подсунув под пострадавшего сухую доску), чем стараться разжать его руки, соблюдая, однако, при этом вышеуказанные меры предосторожности как по отношению к себе, так и по отношению к пострадавшему. В случае необходимости следует перерубить или перерезать провода низкого напряжения соответствующим изолированным инструментом. Производить это нужно с должной осторожностью, не касаться проводов, резать каждый провод в отдельности.

***При высоком напряжении***

Для отделения пострадавшего от земли или от токоведущих частей следует надеть перчатки и действовать штангой или клещами на соответствующее напряжение. При напряжении до 10 кВ, если пострадавший касается одного полюса или одной фазы (ток идет через тело в землю), можно с помощью указанных приспособлений пододвинуть ему под ноги сухую доску или другое изолирующее приспособление. На линиях электропередачи, когда освобождение пострадавшего от тока одним из указанных выше способов не может быть осуществлено достаточно быстро и безопасно, необходимо прибегнуть к замыканию накоротко (наброс и т.д.) всех проводов линии и к надежному заземлению их (согласно общим правилам техники безопасности). При этом должны быть приняты меры, чтобы набрасываемые провода не коснулись тела спасающего. Если пострадавший находится на

высоте, надо предупредить или обезопасить его падение. Если пострадавший касается одного провода, то часто достаточно заземлить только один провод. Осуществляя заземление и закорачивание, следует применяемый для этого провод сначала соединить с землей, а затем набросить его на линейные провода, подлежащие заземлению. Необходимо также помнить, что и по отключении высоковольтной линии на ней, в случае достаточной емкости, может сохраниться опасный для жизни заряд и что лишь надежное заземление линии может ее обезопасить.

### *1.4 Меры первой помощи*

Успех оказания первой помощи зависит от быстрого действия, находчивости и умения подающих помощь. Оживление пострадавшего зависит в большинстве случаев от быстроты освобождения его от тока и быстрого перехода к правильному и безостановочному производству искусственного дыхания.

Период клинической смерти продолжается в лучшем случае 7-8 минут, поэтому при смертельном поражении током следует немедленно приступить к оказанию пострадавшему медицинской помощи с целью восстановления его жизненных функций. В случае, если пострадавший не дышит или дышит очень слабо, промедление и долгие сборы влекут за собой гибель пострадавшего. Меры первой помощи зависят от того состояния, в котором находится пострадавший после освобождения его от тока.

Если пострадавший в сознании, но до этого был в обморочном состоянии или продолжительное время находился под током, то ввиду возможного ухудшения необходимо все же направить его или доставить к врачу. При тяжелом состоянии обязательно вызвать врача (скорую помощь).

При бессознательном состоянии пострадавшего надо уложить удобно, ровно, покойно. Распустить, расстегнуть одежду, создать приток свежего воздуха, удалить лишних людей. Давать нюхать нашатырный спирт, брызгать водой (не изо рта), растирать и согревать тело. Срочно вызвать врача. Если пострадавший плохо дышит – очень редко и судорожно, необходимо делать искусственное дыхание и массаж сердца.

При отсутствии признаков жизни (дыхания, сердцебиения, пульса) нельзя все же считать пострадавшего мертвым. Смерть часто бывает лишь кажущейся. В таком случае должна быть немедленно оказана первая помощь в виде искусственного дыхания. Искусственное дыхание необходимо делать непрерывно до прибытия врача.

Приступая к оживлению пострадавшего, одновременно с вызовом врача на место происшествия следует:

- ✓ освободить потерпевшего от всех стесняющих его одежд;
- ✓ обеспечить доступ чистого воздуха;
- ✓ удалить лишних людей;
- ✓ быстро, не теряя времени, освободить рот потерпевшего от слизи, крови и т.п.; если рот стиснут и легко не раскрывается, нужно, надавив пальцем на нижнюю челюсть, выдвинуть ее вперед так, чтобы нижние зубы стали впереди верхних, и разжать пострадавшему рот, чтобы удержать рот в разжатом состоянии, следует взять какой-либо продолговатый предмет, хотя бы кусок дерева, и втиснуть между зубами; язык, если он глубоко запал, следует вытянуть, захватив носовым платком.

### *1.5 Приемы искусственного дыхания*

Основным приемом искусственного дыхания является дыхание «рот в рот» и непрямой массаж сердца. Способ искусственного дыхания «рот в рот» заключается в том, что оказывающий помощь производит выдох из своих легких в легкие пострадавшего непосредственно в рот или в нос пострадавшего или через специальное приспособление в рот. Для этого необходимо быстро раскрыть у пострадавшего рот, удалить из него посторонние предметы и слизь, запрокинуть ему назад голову и оттянуть нижнюю челюсть. После этого оказывающий помощь делает глубокий вдох и с силой выдыхает воздух в рот пострадавшего. При вдувании воздуха оказывающий помощь плотно прижимает свой рот к лицу пострадавшего так, чтобы по возможности охватить своим ртом весь рот пострадавшего, а своим лицом зажать ему нос, чтобы вдыхаемый воздух не

выходил бы через нос. После этого спасающий откидывается назад и делает новый вдох. В этот период грудная клетка пострадавшего опускается, и он непроизвольно делает пассивный выдох.

При невозможности полного охвата рта пострадавшего вдувать воздух в легкие следует через нос, плотно закрыв при этом рот пострадавшего. Вдувание воздуха в рот или нос можно производить через марлю, салфетку или носовой платок, следя за тем, чтобы при каждом вдувании происходило достаточное расширение грудной клетки пострадавшего. При возобновлении у пострадавшего самостоятельного дыхания некоторое время следует продолжить искусственное дыхание до полного приведения пострадавшего в сознание или до прибытия врача. В этом случае вдувание воздуха следует производить одновременно с началом собственного вдоха пострадавшего.

При выполнении искусственного дыхания необходимо избегать чрезмерного сдавливания грудной клетки ввиду возможности перелома ребер.

Нельзя также допускать охлаждения пострадавшего (не оставлять его на сырой земле, каменном, бетонном или металлическом полу). Под пострадавшего следует постелить что-либо теплое, а сверху укрыть его.

### 1.6 Наружный массаж сердца

При отсутствии у пострадавшего пульса возможны следующие нарушения деятельности сердца:

✓ резкое ослабление или даже полное прекращение сокращений сердца, что бывает следствием длительного нахождения пострадавшего под действием тока, а также отсутствие своевременной помощи в случае первичного поражения дыхания;

✓ образования под действием электрического тока разрозненных и разновременных (фибриллярных) сокращений отдельных групп волокон сердечной мышцы, которые не могут обеспечить работу сердца в качестве насоса, нагнетающего кровь в сосуды.

Поэтому при отсутствии у пострадавшего пульса одновременно с искусственным дыханием проводится наружный массаж сердца. При этом следует иметь в виду, что без правильной и своевременной предварительной помощи пострадавшему (до прибытия врача) врачебная помощь может оказаться запоздалой и неэффективной.

Наружный массаж производится путем ритмичных сжатий сердца через переднюю стенку грудной клетки при надавливании на относительно подвижную нижнюю часть грудины, позади которой расположено сердце. При этом сердце прижимается к позвоночнику и кровь из его полостей выжимается в кровеносные сосуды. Повторяя надавливание с частотой 60–70 раз в минуту, можно обеспечить достаточное кровообращение в организме при отсутствии работы сердца.

Для проведения массажа сердца пострадавшего следует уложить спиной на жесткую поверхность, обнажить у него грудную клетку, снять стесняющую дыхание одежду. Оказывающий помощь должен стать с левой или правой стороны пострадавшего и занять такое положение, при котором возможен более или менее значительный наклон над пострадавшим. Определив положение нижней трети грудины, оказывающий помощь должен положить на нее верхний край ладони разогнутой до отказа руки, а затем поверх руки положить другую руку и надавливать на грудную клетку пострадавшего, слегка помогая при этом наклоном своего корпуса. Надавливание следует производить быстрыми толчками так, чтобы продвинуть нижнюю часть грудины вниз на 3–4 см, у полных людей на 5–6 см. Следует избегать надавливания на окончание нижних ребер, так как это может привести к их перелому.

Ни в коем случае нельзя надавливать ниже края грудной клетки (на мягкие ткани), так как можно повредить расположенные здесь органы. Надавливание на грудину следует производить 1 раз в секунду. После быстрого толчка руки остаются в достигнутом положении примерно в течение 1/3 с. После этого руки следует снять и освободить грудную клетку от давления. При сочетании искусственного дыхания с массажем сердца следует после двух-трех глубоких дыханий в рот или в нос пострадавшего производить 15–20 надавливаний на грудную клетку и так чередовать.

При правильном проведении искусственного дыхания и массажа сердца у пострадавшего появляются следующие признаки оживления:

✓ улучшение цвета лица, приобретающего розовый оттенок вместо серо-землистого цвета с синеватым оттенком, который был у пострадавшего до оказания помощи;

✓ появление самостоятельных дыхательных движений, которые становятся все более равномерными по мере продолжения мероприятий по оказанию помощи;

✓ сужение зрачков; степень сужения зрачков может служить наиболее верным показателем эффективности оказываемой помощи: узкие зрачки у оживляемого указывают на достаточное снабжение мозга кислородом.

Описанный выше прием искусственного дыхания весьма эффективен, однако он требует помощника. Кроме того, им нельзя пользоваться при переломе ребер у пострадавшего. В этом случае пользуются вторым способом. Освобождают пострадавшего до пояса от одежды, раскрывая рот, вытаскивают язык, захватив его платком. Во время искусственного дыхания по этому способу необходимо следить за языком, чтобы он не запал и не закрыл доступ воздуха в дыхательные пути.

Пострадавшего кладут на спину, подложив под лопатки сверток одежды, чтобы голова запрокинулась назад. Производящий искусственное дыхание располагается у головы пострадавшего, захватив его руки за предплечья у локтя, и заносит их за голову пострадавшего. Этим достигается расширение грудной клетки, и внешний воздух устремляется в легкие, происходит наполнение легких воздухом (вдох), затем руки пострадавшего опускаются и придавливаются к обеим сторонам груди, чем достигается сжатие грудной клетки. При этом воздух из легких устремляется наружу (выдох). Искусственное дыхание производится спокойно, не более 12–15 раз в минуту. Этот способ неприемлем, если у пострадавшего сломана рука или ключица.

**2 Вводная часть**

**Цель работы:** опытное исследование свойств линейной электрической цепи, нахождение токов в ветвях методом наложения и по законам Кирхгофа, определение потенциалов точек электрической цепи, исследование передачи энергии от активного двухполюсника нагрузке и сопоставления опытных и теоретических данных.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

✓ заглавие (номер и название лабораторной работы);

✓ схемы исследованных электрических цепей;

✓ результаты исследований (в виде таблиц и графиков);

✓ расчетную часть задания;

✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Набор элементов и приборов, используемых в лабораторной работе***

Элементы, используемые в данной лабораторной работе, размещены на первой панели МЭЛ (рисунок 1).

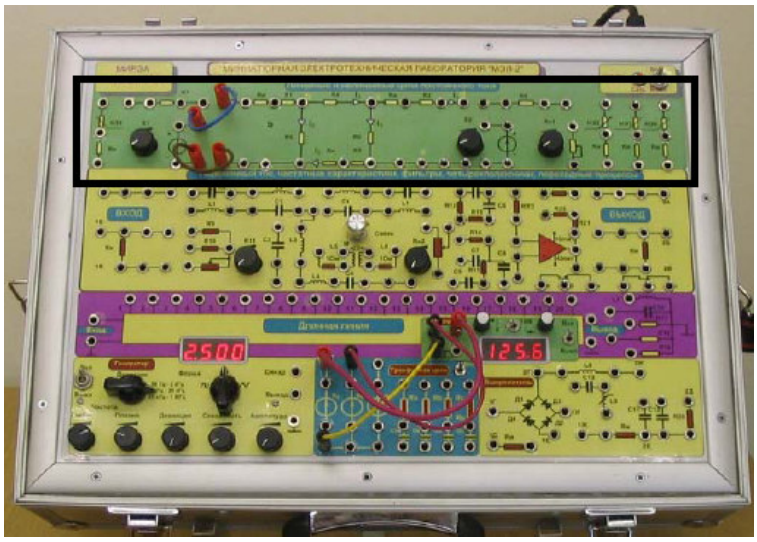


Рисунок 1. Панель МЭЛ для моделирования линейных и нелинейных цепей постоянного тока

**3 Теоретические сведения и методы численного расчета линейных электрических цепей**

В электрических цепях постоянного тока источниками энергии являются источники постоянного напряжения E и источники постоянного тока J. Понятие «постоянное напряжение (ток)» означает, что во времени значение и направление напряжения (тока) не меняются. Можно сказать, что частота изменения постоянного напряжения (тока) $\omega = 0$.

В идеальном источнике напряжения внутреннее сопротивление $R_{ин} = 0$ и напряжение на выходе идеального источника напряжения равно E и не зависит от внешней цепи.

В идеальном источнике тока внутреннее сопротивление $R_{ит} = \infty$, ток, отдаваемый во внешнюю цепь, не зависит от свойств внешней цепи.

Источник тока J с параллельно включенным сопротивлением $R_{ит}$ можно заменить эквивалентным источником напряжения $E_2 = J * R_{ит}$, в котором $R_{ин} = R_{ит}$ (рисунок 2).

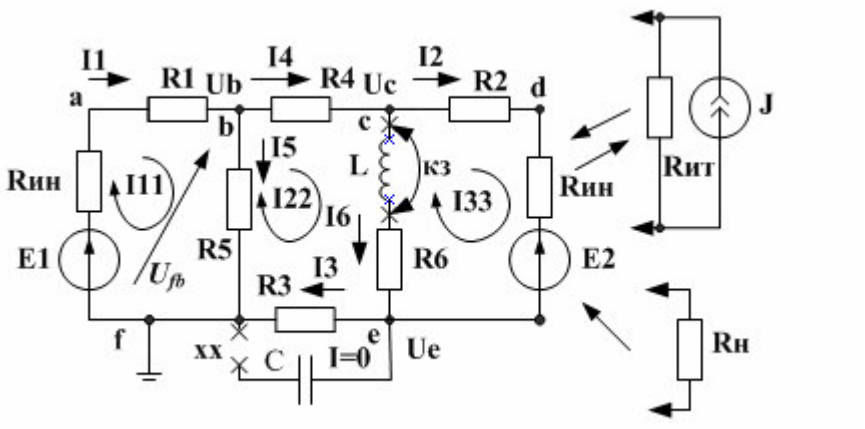


Рисунок 2. Схема линейной цепи постоянного тока

И наоборот, источник напряжения $E_2$ с внутренним сопротивлением $R_{ин}$ можно заменить источником тока $E = \frac{E_2}{R_{ин}}$, в котором $R_{ин} = R_{ит}$.

Пассивными элементами электрической цепи являются сопротивление (резистор), индуктивность (катушка индуктивности), емкость (конденсатор).

По закону Ома напряжение на резисторе $u_R = i * R$.

В цепи постоянного тока (i = const, u = const ) напряжение на индуктивности

$$u_L = L\frac{di_L}{dt} = 0$$

и индуктивность эквивалентна короткому замыканию (КЗ).

Ток через емкость

$$i_C = C\frac{du_c}{dt} = 0$$

и емкость эквивалентна разрыву (холостому ходу – ХХ).

Вольтамперной характеристикой элемента электрической цепи называют зависимость тока, проходящего через этот элемент, от напряжения на его зажимах.

Электрическая цепь постоянного тока является линейной, если все элементы цепи имеют линейные вольтамперные характеристики**.**

Структура электрической цепи определяется взаимным расположением ветвей, узлов и контуров. Ветвь – это участок цепи, через который проходит один и тот же ток. Узел – место соединения трех и более ветвей. Контур – замкнутый путь, последовательность ветвей и узлов, в которой каждая ветвь и каждый узел входит один раз.

***Основные законы электрических цепей***

А. Обобщенный закон Ома для участка цепи, содержащего источник напряжения: ток в ветви равен напряжению на зажимах ветви, взятому по направлению тока, плюс (минус) источники напряжения, деленному на сумму сопротивлений ветви.

Ток в первой ветви (см. рисунок 2)

$$I_1 = \frac{U_{fb} + E_1}{R_{ин} + R_1}.$$

Знак плюс берут для источников напряжения, совпадающих по направлению с током.

В. Первый закон Кирхгофа: алгебраическая сумма токов, сходящихся в узле, равна нулю (или сумма входящих в узел токов равна сумме выходящих токов).

Для узла b (см. рисунок 2): $I_1 = I_4 + I_5$.

С. Второй закон Кирхгофа: в замкнутом контуре алгебраическая сумма падений напряжений на пассивных элементах равна алгебраической сумме источников напряжения. При этом со знаком плюс берут падения напряжения на тех пассивных элементах, в которых токи совпадают с направлением обхода контура. Со знаком плюс берут источники напряжения, совпадающие по направлению с направлением обхода контура.

***Алгебраические методы расчета электрических цепей***

Для расчета сложных электрических цепей применяют алгебраические методы, основанные на составлении и решении систем уравнений для токов и напряжений в цепи. Рассмотрим применение алгебраических методов для расчета цепи, показанной на рисунке 2.

А. Расчет цепи по уравнениям Кирхгофа.

Записываем уравнения по первому закону Кирхгофа для узлов b, c, e:

$$\begin{aligned} &Узел\,b:\ I_1 = I_4 + I_5;\\ &Узел\,c:\ I_4 = I_2 + I_6;\\ &Узел\,e:\ I_3 = I_6 + I_2. \end{aligned} \qquad (1)$$

Записываем уравнения по второму закону Кирхгофа для трех контуров. Для этого последовательно обходим контуры по направлению обхода и приравниваем сумму падений напряжений на пассивных элементах контура сумме ЭДС:

$$\begin{aligned} &1\text{-й контур}: R_{вн}I_1 + R_1I_1 + R_5I_5 = E_1;\\ &2\text{-й контур}: R_4I_4 + R_6I_6 + R_3I_3 - R_5I_5 = 0;\\ &3\text{-й контур}: R_2I_2 + R_{вн}I_2 - R_6I_6 = -E_2. \end{aligned}$$

(2)

B. Расчет цепи методом контурных токов.

Независимые контуры и контурные токи $I_{11}$, $I_{22}$, $I_{33}$ обозначены на схеме (см. рисунок 2).

Записываем канонические уравнения по методу МКТ для трехконтурной схемы:

$$\begin{pmatrix} I_{11}\\ I_{22}\\ I_{33} \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13}\\ R_{21} & R_{22} & R_{23}\\ R_{31} & R_{32} & R_{33} \end{pmatrix}^{-1} \cdot \begin{pmatrix} E_{11}\\ E_{22}\\ E_{33} \end{pmatrix}.$$

(3)

Диагональные сопротивления контурной матрицы сопротивлений с одинаковыми индексами находим как сумму всех сопротивлений контура при последовательном обходе. Недиагональные сопротивления с разными индексами равны сопротивлениям смежных ветвей контуров, причем со знаком плюс берут те сопротивления смежных ветвей, в которых контурные токи направлены одинаково. Контурные ЭДС равны алгебраической сумме всех ЭДС контура. Со знаком плюс берут ЭДС, совпадающие по направлению с обходом контура.

C. Расчет методом узловых напряжений.

Составляем уравнения по методу узловых напряжений.

Нумеруем узлы: b→1, c→2, e→3.

Записываем для схемы с тремя независимыми узлами канонические уравнения метода узловых напряжений для расчета напряжений в узлах:

$$\begin{pmatrix} U_1\\ U_2\\ U_3 \end{pmatrix} = \begin{pmatrix} G_{11} & G_{12} & G_{13}\\ G_{21} & G_{22} & G_{23}\\ G_{31} & G_{32} & G_{33} \end{pmatrix}^{-1} \cdot \begin{pmatrix} J_{11}\\ J_{22}\\ J_{33} \end{pmatrix}.$$

(4)

В матрице узловых проводимостей диагональные проводимости с одинаковыми индексами находим как сумму проводимостей всех ветвей, подключенных к узлу с таким же индексом. Недиагональная проводимость с разными индексами (например $G_{12}$) равна взятой со знаком минус проводимости всех ветвей, соединяющих узлы 1 и 2.

Узловые токи находим как алгебраическую сумму подключенных к данному узлу источников напряжения, деленных на сопротивления ветвей, в которых они находятся, и источников тока. Со знаком плюс берем источники, направленные к узлу.

Матрица узловых проводимостей симметрична относительно главной диагонали: $G_{12} = G_{21}$, $G_{13} = G_{31}$ и т.д. Поэтому при составлении выражений для узловых проводимостей можно использовать это свойство.

D. Метод эквивалентного генератора.

Если требуется найти ток только в одной ветви сложной цепи, применяют метод эквивалентного генератора. Суть его состоит в следующем.

1. Размыкаем ветвь с неизвестным током и находим напряжение холостого хода $U_{xx}$ на зажимах разомкнутой ветви и входное сопротивление $R_{вх}$.

2. Заменяем сложную цепь активным двухполюсником с источником напряжения $E_{экв} = U_{xx}$ и внутренним сопротивлением $R_{вх}$ (рисунок 3).

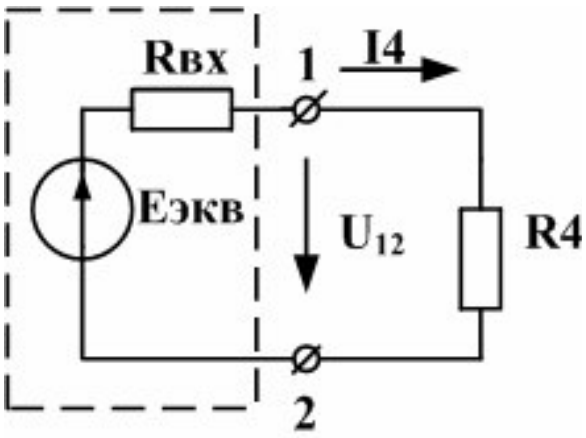



Рисунок 3. Схема эквивалентного генератора с нагрузкой

Это и есть эквивалентный генератор. Подключаем к эквивалентному генератору сопротивление ветви (нагрузки) и вычисляем ток по формуле

$$I = \frac{E_{экв}}{R_{вх} + R_{н}}. \qquad (5)$$

Если $R_н = 0$, ток в ветви называют током короткого замыкания

$$I_{кз} = U_{xx} / R_{вх}.$$

Отсюда видно, что входное сопротивление можно рассчитать как

$$R_{вх} = U_{xx} / I_{кз}.$$

E. Согласование нагрузки с генератором.

Для выделения максимальной мощности в нагрузке требуется выполнить условие согласования нагрузки с генератором $R_н = R_{вх}$.

При этом максимальная мощность в нагрузке равна

$$P_{\max} = \frac{(E_{экв})^2 \cdot R_{н}}{(R_{вх} + R_{н})^2} = \frac{(E_{экв})^2}{4R_{вх}}. \qquad (6)$$

F. Метод наложения.

Метод наложения заключается в том, что ток в какой-либо ветви схемы можно представить как алгебраическую сумму токов, получающихся от отдельных источников напряжения при исключении остальных и сохранении внутреннего сопротивления исключенных источников. Ток в любой (например *m*-й) ветви можно выразить в виде суммы

$$I_m = E_1 g_{m1} + E_2 g_{m2} + \ldots + E_m g_{mm} + \ldots + E_n g_{mn}. \qquad (7)$$

Для опытного определения входной проводимости $g_{mm}$ и взаимных проводимостей $g_{m1}$ и $g_{mn}$ в схеме следует оставить только один источник $E_m$, а остальные источники исключить, оставив их внутренние сопротивления, и измерить токи в ветвях, тогда:

$$g_{m1} = g_{1m} = \frac{I_1}{E_m}; g_{mm} = \frac{I_m}{E_m}; g_{mn} = g_{nm} = \frac{I_n}{E_m}.$$

(8)

**4 Практическая часть**

На первой панели МЭЛ (см. рисунок 1) имеется электрическая цепь (рисунок 4), состоящая из постоянных сопротивлений R1, R2, R3, R4, R5, R6 и трех измерительных сопротивлений Rи = 10 Ом. Значения постоянных сопротивлений следующие: R1 = 100 Ом, R2 = 200 Ом, R3 = 270 Ом, R4 = 300 Ом, R5 = 150 Ом, R6 = 360 Ом.

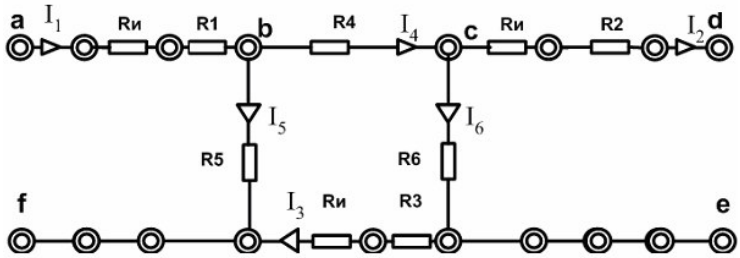



Рисунок 4. Схема реального моделирования цепи

Исследуемая линейная цепь собирается подключением к пассивной цепи (см. рисунок 4) источников напряжения E1, E2 и переменного сопротивления нагрузки Rн (рисунок 5).

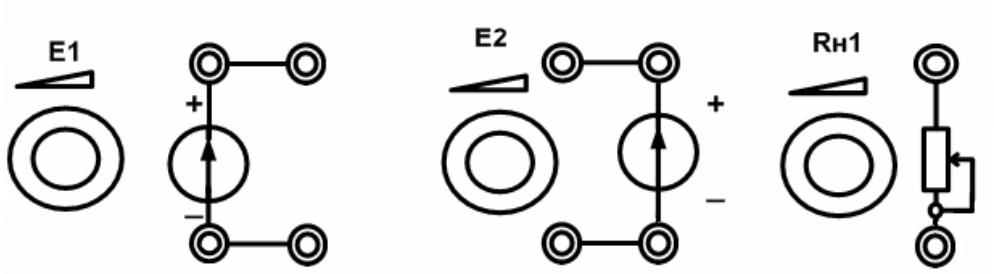



Рисунок 5. Источники напряжения и сопротивление нагрузки

Измерительные сопротивления Rи включены в первую, вторую и третью ветви схемы. Измерения проводятся мультиметром или вольтметром постоянного тока. Токи в остальных ветвях находят по первому закону Кирхгофа.

Численные значения напряжений источников задаются преподавателем.

***Лабораторное задание***

1. Установите заданные преподавателем величины напряжения каждого из двух источников напряжения. Измерения проводите вольтметром. Значения напряжений запишите в рабочий отчет и поддерживайте неизменными.

2. Подключите к гнездам a, f цепи (см. рисунок 4) источник напряжения $E_1$. Гнезда d, e замкните перемычкой.

3. Определите токи во всех ветвях при действии только источника напряжения $E_1$. Результаты запишите в таблицу 1.

Таблица 1

| | E1 ≠ 0<br>E2 = 0 | E1 = 0<br>E2 ≠ 0 | E1 ≠ 0<br>E2 = 0<br>(опыт) | E1 ≠ 0<br>E2 = 0<br>(расчет) |
|---|---|---|---|---|
| $I_1$, мА | | | | |
| $I_2$, мА | | | | |
| $I_3$, мА | | | | |

Для определения токов в реальной модели можно использовать мультиметр в режиме вольтметра, подключая его параллельно измерительным сопротивлениям Rи = 10 Ом в первой, второй и третьей ветвях. Значение тока в ветви рассчитывается по формуле

$$I_k = \frac{U_{kИ}}{10\ \text{Ом}},$$

где $U_{kИ}$ – напряжение, измеренное на измерительном сопротивлении в ветви с номером 1, 2, 3; $I_k$.– ток в этой ветви.

Токи надо определять с учетом знаков для условно положительных направлений, обозначенных на схеме (см. рисунок 4). Для этого "плюс" прибора следует подключать к тому гнезду измерительного сопротивления, к которому направлена стрелка тока.

Токи в остальных ветвях рассчитайте по первому закону Кирхгофа.

4. Подключите к гнездам d, e источник напряжения $E_2$, а вместо источника напряжения $E_1$ гнезда a, f замкните перемычкой. Так же, как в п. 3, определите токи во всех ветвях (с учетом их знаков) при действии только источника напряжения $E_2$. Результаты запишите в таблицу 1.

5. Включите два источника напряжения. Определите токи в ветвях схемы при действии обоих источников (с учетом знаков токов). Результаты запишите в таблицу 1.

6. По опытным данным пунктов 3 и 4 подсчитайте токи во всех ветвях при действии обоих источников напряжения и сравните результаты с экспериментом п. 5.

7. В схеме с одним источником напряжения $E_1$, подключенным к гнездам a, f , измерьте напряжение холостого хода Uxx между разомкнутыми точками d, e и ток короткого замыкания Iкз при замкнутых зажимах (гнездах) d, e. В реальной модели ток короткого замыкания определите по напряжению на измерительном сопротивлении во второй ветви. Рассчитайте входное сопротивление цепи со стороны правых зажимов:

$$R_{BX} = \frac{U_{XX}}{I_{K3}}.$$

8. Исключите два источника напряжения. Измерьте мультиметром в режиме омметра входное сопротивление со стороны правых зажимов. При измерении входного сопротивления вместо удаленных источников напряжения и тока надо оставить их внутренние сопротивления. Сравните результаты, полученные в п.п. 7 и 8.

9. Включите два источника напряжения и измерьте мультиметром напряжения во всех узлах схемы относительно узла f. Результаты запишите в таблицу 2.

Таблица 2

| $\varphi_a$ | $\varphi_b$ | $\varphi_c$ | $\varphi_d$ | $\varphi_e$ |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| | | | | |

10. Включите вместо источника напряжения $E_2$ переменное сопротивление нагрузки Rн. Изменяя значение Rн от нуля (режим кз) до ∞ (в режиме хх), измерьте в 6–7 точках напряжение на нагрузке и ток в ней. Одно из значений Rн должно равняться входному сопротивлению, измеренному в п. 7. Запишите показания в таблицу 3. Рассчитайте мощность, выделяемую в нагрузке, и сопротивление Rн. Найдите максимальное значение мощности и соответствующие максимальной мощности значения тока и сопротивления нагрузки $R_{Н_{ОПТ}}$. Проверьте выполнение условия согласования нагрузки с генератором.

Таблица 3

| № п/п | $I_1$ мА | $I_2$ мА | $I_3$ мА | $U_2$ В | $R_н$ Ом | $P_2$ Вт |
|---|---|---|---|---|---|---|
| | | | | | | |

11. По опытным данным пункта 5 рабочего задания подсчитать токи во всех ветвях при действии обоих источников напряжения.

12. По данным из таблицы 1 подсчитать входные и взаимные проводимости ветвей $g_{11}$, $g_{12}$, $g_{21}$, $g_{22}$ и записать выражения для токов $I_1$ и $I_2$ по принципу наложения.

13. Подсчитать, при каком значении напряжения $E_2$ ток во второй ветви будет равен нулю.

14. По данным таблицы 2 и значениям токов рассчитать сопротивления всех ветвей схемы. Сравнить результаты с заданными по схеме.

15. Рассчитать входное сопротивление схемы со стороны правых зажимов. Сравнить с результатами предыдущих измерений и расчетов.

**Лабораторная работа № 2. Исследование цепей переменного тока**

***Продолжительность: 180 минут.***
***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 1.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

**Цель работы:** исследование амплитудных и фазовых соотношений в цепях переменного тока, частотных характеристик и резонансных явлений, построение векторных диаграмм токов и напряжений.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один

рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Набор элементов и приборов, используемых в лабораторной работе***

Пассивные элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

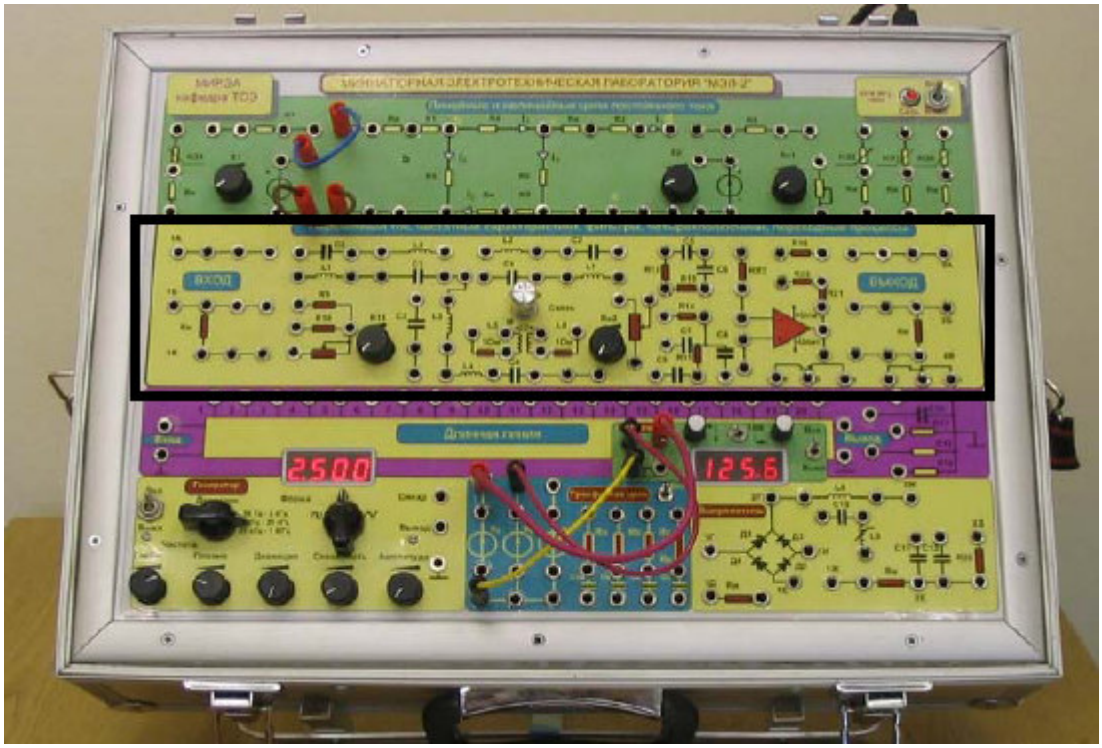



Рисунок 1. Панель МЭЛ для моделирования цепей переменного тока

## 2 Теоретические сведения и методы численного расчета электрических цепей переменного тока

В электрических цепях переменного тока токи и напряжения меняются во времени и могут иметь синусоидальную гармоническую форму или периодическую несинусоидальную форму. Поэтому электрические цепи переменного тока разделяют на цепи синусоидального тока и цепи несинусоидального тока.

При гармоническом синусоидальном сигнале $e(t) = E_m * \sin(\omega t + \psi_E)$ расчет электрических цепей проводят символически методом с использованием комплексных амплитуд токов и напряжений и комплексных сопротивлений.

Рассмотрим пример расчета простой цепи синусоидального тока (рисунок 2).

Для расчета символическим методом исходную цепь для мгновенных значений напряжений и токов (см. рисунок 2) заменяют символической схемой замещения для комплексных амплитуд напряжений и токов и комплексных сопротивлений (рисунок 3).

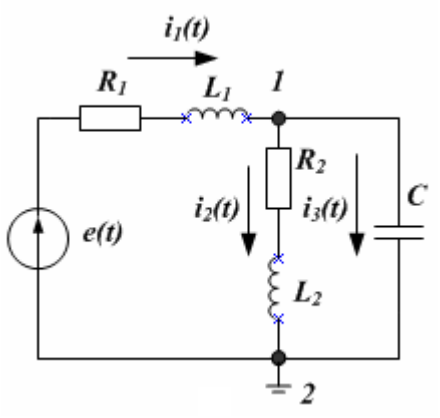



Рисунок 2. Схема простой цепи синусоидального тока

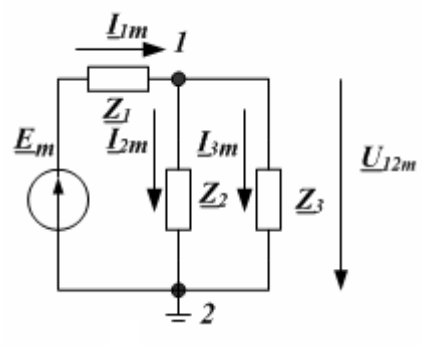



Рисунок 3. Символическая схема замещения для комплексных амплитуд напряжений и токов и комплексных сопротивлений

В символической схеме замещения комплексная амплитуда входного напряжения

$$\underline{E}_m = E_m \cdot e^{j\psi}.$$

Сопротивление каждой ветви цепи характеризуют комплексным сопротивлением

$$\underline{Z} = R + jX = R + j(\omega L - \frac{1}{\omega C}) = Z \cdot e^{j\varphi}, \quad (1)$$

где $Z = \sqrt{R^2 + X^2}$ – модуль комплексного сопротивления; $\varphi = \operatorname{arc\,tg}\left(\frac{X}{R}\right)$ – аргумент комплексного сопротивления; R – активное сопротивление ветви; $X = (\omega L - \frac{1}{\omega C})$ – реактивное сопротивление ветви.

Часть цепи, содержащая одну или несколько ветвей и имеющая два входных зажима, называется двухполюсником. Входное эквивалентное сопротивление двухполюсника рассчитывают сверткой цепи. Например, для схемы, изображённой на рисунке 3:

$$\underline{Z}_{\text{экв}} = \underline{Z}_1 + \frac{\underline{Z}_2 \cdot \underline{Z}_3}{\underline{Z}_2 + \underline{Z}_3}.$$

Входной ток

$$\underline{I}_{1m} = \frac{\underline{E}_m}{\underline{Z}_{\text{экв}}} = \frac{E_m \cdot e^{j\psi_E}}{Z_{\text{экв}} \cdot e^{j\varphi}} = I_{1m}(\omega) \cdot e^{j\psi_I(\omega)}.$$

Здесь зависимость амплитуды тока от частоты $I_{1m}(\omega)$ – амплитудно-частотная характеристика тока, $\psi_I(\omega) = \psi_E(\omega) - \varphi(\omega)$ – фазочастотная характеристика тока. Если принять $\psi_E = 0$, то $\psi_I(\omega) = -\varphi(\omega)$. В цепи с индуктивным сопротивлением $[-\varphi(\omega)]$ меньше нуля и напряжение опережает ток по фазе. В цепи с емкостным сопротивлением $[-\varphi(\omega)]$ больше нуля и напряжение отстает от тока по фазе. В цепи с чисто активным сопротивлением, а также в резонансных режимах, когда $X_{\text{экв}} = 0$, ток совпадает с напряжением по фазе. Мгновенное значение тока на входе двухполюсника (см. рисунок 2) равно $i_1(t) = I_{1m} * \sin(\omega t + \psi_I)$.

Мгновенная мощность будет равна

$$p(t) = u(t) \cdot i_1(t) = \frac{U_m \cdot I_{1m}}{2}\cos\varphi - \frac{U_m \cdot I_{1m}}{2}\cos(2\omega t + 2\psi_U - \varphi).$$

(2)

В этой формуле u(t) = e(t) – напряжение на входе двухполюсника.

Средняя мощность за период или активная мощность

$$P = \frac{1}{T}\int_0^T u \cdot i \cdot dt = UI\cos\varphi.$$

Здесь $U = \frac{U_m}{\sqrt{2}}$ и $I = \frac{I_m}{\sqrt{2}}$ – действующие значения напряжения и тока на входе двухполюсника.

В расчетах символическим методом применяют комплексную мощность

$$\tilde{S} = \underline{U} \cdot \underline{I}^* = P + jQ,$$

где $\underline{U}$ – комплексное действующее значение напряжения на входе пассивного двухполюсника; $\underline{I}^*$ – комплексно-сопряженный ток; P – активная мощность; Q – реактивная мощность.

В цепи синусоидального тока, содержащей элементы R, L, C, токи в ветвях могут существенно отличаться по фазе от входного напряжения. Поэтому расчет можно выполнить, только используя комплексные амплитуды и комплексные сопротивления.

Для расчета комплексных амплитуд напряжений и токов в символической схеме замещения цепи можно применять любые методы расчета линейных цепей, изученные для цепей постоянного тока. От комплексных амплитуд легко перейти к функциям времени для мгновенных значений.

## 3 Практическая часть

Номинальные значения индуктивностей и емкостей указаны в спецификации стендов МЭЛ и составляют: $L_1 = L_3 = 10$ мГн, $L_2 = L_4 = 6{,}8$ мГн, $C_1 = C_3 = 47$ нФ, $C_2 = C_4 = 68$ нФ. Резисторы: $R_9 = 1$ кОм, $R_{10} = 100$ Ом, измерительные резисторы $R_и = 10$ Ом, переменный резистор $R_{11} = 2{,}2$ кОм.

Гнезда "Вход" и "Выход" можно использовать для подключения приборов.

В лабораторной работе используются следующие приборы: функциональный генератор сигналов (ГС), электронный вольтметр переменного тока (V), двухканальный осциллограф (Осц), фазометр (Ф).

***Лабораторное задание***

А. Исследование RL- и RC-цепей.

1. Поставить переключатели чувствительности вольтметра в положение 1В. Установить минимальное значение входного сигнала, повернув ручку "Амплитуда" ГС против часовой стрелки до упора. Включить питание вольтметра, генератора сигналов, фазометра, осциллографа.

2. Установить на генераторе: форму сигнала – синусоидальную; диапазон частот "50 Гц – 2 кГц"; регулировку девиации частоты установить в крайнее левое положение (со щелчком).

3. Собрать на макете схему, показанную на рисунке 4, используя гнезда "Вход" и "Выход" стенда МЭЛ для подключения приборов.

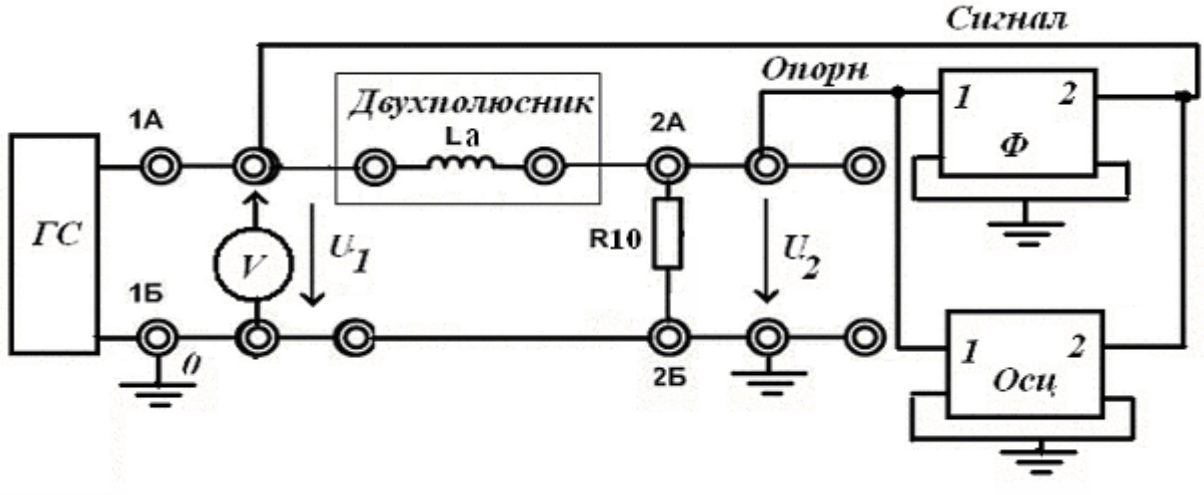



Рисунок 4. Схема реального моделирования цепи

4. Двухполюсники, которые должны быть исследованы в работе, показаны на рисунке 5. При исследованиях эти цепи включают между точками 1А и 2А схемы рисунка 4. Значения индуктивностей La, Lb и емкостей Ca, Cb выбираются из таблицы 1 согласно указаниям преподавателя.

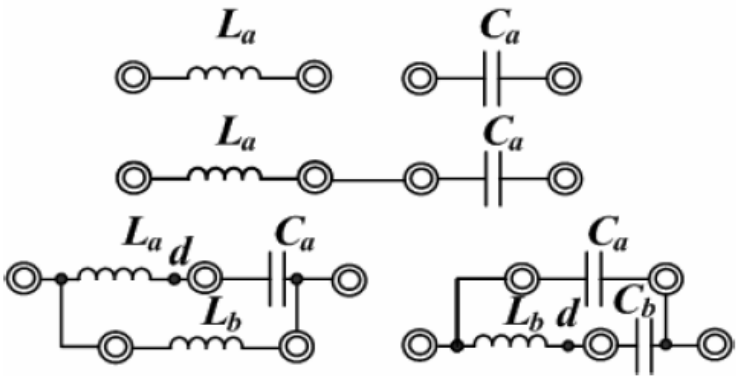



Рисунок 5. Схемы двухполюсников

Таблица 1

| № бригады | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $L_a$ | $L_1$ | $L_1$ | $L_2$ | $L_2$ | $L_2$ | $2L_1$ | $2L_1$ | $L_2$ | $2L_1$ | $2L_2$ | $L_1$ | $L_1$ |
| $L_b$ | $L_1$ | $L_1$ | $L_1$ | $L_1$ | $L_2$ | $L_2$ | $L_2$ | $L_2$ | $L_2$ | $2L_1$ | $2L_2$ | $L_2$ |
| $C_a$ | $C_1$ | $C_2$ | $C_1$ | $C_2$ | $0{,}5C_2$ | $0{,}5C_1$ | $C_2$ | $0{,}5C_1$ | $C_1$ | $C_1$ | $2C_1$ | $0{,}5C_1$ |
| $C_b$ | $C_1$ | $C_2$ | $C_1$ | $C_2$ | $C_1$ | $C_1$ | $C_2$ | $C_2$ | $C_1$ | $C_2$ | $C_2$ | $0{,}5C_2$ |

5. Подключить между точками 1А и 2А индуктивность La и рассчитать частоту сигнала генератора $f_1$, при которой реактивное сопротивление индуктивности равно сопротивлению $R_{10}$: $X_L = 2\pi f_1 La = R_{10}$.

6. Подключить вольтметр V к гнезду 1А, установить выходное напряжение ГС, равное 1 В, а частоту ГС, равной 0,1 $f_1$.

7. Входы $U_0$ и $U_c$ фазометра стенда МЭЛ подключить к выходу генератора, поставить переключатель в положение «$0^o$» и регулировкой «Установка нуля» получить нулевое показание фазометра. Затем поставить переключатель в положение «$180^o$» и установить на цифровой шкале фазометра показание «+/-180».

8. Поставить переключатель фазометра в положение «$0^o$». Переключить вход $U_0$ фазометра на точку 2А к сопротивлению $R_{10}$, вход $U_1$ оставить подключенным к сигнальному выходу ГС (гнездо 1А). При этом фазометр измеряет разность фаз между напряжением и током, причем опорным сигналом является напряжение на $R_{10}$, пропорциональное току. При отсутствии фазометра разность фаз надо рассчитывать, измерив для каждой частоты временной сдвиг осциллограммы сигнального напряжения относительно опорного напряжения, пропорционального току. Сдвиг фаз принято отсчитывать от тока к напряжению. Разность фаз рассчитывается по формуле
$\varphi = 2\pi f(T2-T1)$, где (Т2 – Т1) – разность ближайших моментов пересечения с положительной производной нулевого уровня, соответственно, опорным и сигнальным напряжением.

9. Изменяя частоту ГС ручками «Частота грубо» и "Частота плавно" в пределах от $0{,}1f_1$ до $10f_1$, измерить вольтметром для 10 точек напряжения в точках 1А и 2А и показания фазометра Ф. Наблюдать форму сигналов и временной сдвиг на осциллографе. Зарисовать в масштабах осциллограммы. Результаты записать в таблицу 2.

Таблица 2

Цепь RL

| N | f, кГц | $U_1$ | $U_2$ | I | T2-T1 | φ |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |

10. Включить между точками 1А и 2А емкость Са и рассчитать частоту сигнала генератора $f_2$, при которой реактивное сопротивление емкости равно сопротивлению $R_{10}$:

$$X_C = \frac{1}{2\pi f_2 C_a} = R_{10}\ .$$

Изменяя частоту ГС в пределах 0,1 $f_2$ до 10 $f_2$, повторить измерения п. 9. Результаты занести в таблицу, аналогичную таблице 2.

В. Исследование RLC-цепи.

1. Включить между точками 1А и 2А последовательно индуктивность La и емкость Са. Изменяя частоту ГС, найти максимальное напряжение на последовательном сопротивлении $R_{10}$. Частота сигнала при этом будет близка к резонансной, а разность фаз между током и напряжением в цепи близка к нулю. Записать значение резонансной частоты $f_p$. Измерить значения напряжений $U_1$ и $U_2$ и фазы при изменении частоты от 2 кГц до 20 кГц. Результаты занести в таблицу, аналогичную таблице 2.

С. Исследование трехэлементного двухполюсника.

1. Подключить к точкам 1А и 2А одну из схем трехэлементных реактивных двухполюсников. Схемы с двумя индуктивностями и с двумя емкостями исследуются бригадами согласно указаниям преподавателя.

Установить входное напряжение ГС равным 0,1В. Измерить значения напряжений $U_1$, $U_2$ и фазы $\varphi_{1A}$ в диапазоне частот от 2 до 20 кГц и результаты записать в таблицу 3.

Таблица 3

| N | f, кГц | $U_1$ | $U_2$ | I | T2-T1 | $\varphi$ | $\underline{Z}$вх | $\lvert\underline{Z}$вх$\rvert$ |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| | | | | | | | | |

2. Для схемы с трехэлементным двухполюсником на частоте f = N + 2 кГц (N – задается преподавателем) измерить напряжения и фазы в точках схемы 1A, 2A, d. Результаты записать в таблицу 4.

Таблица 4

| $U_{1A}$ | $\varphi_{1A}$ | $U_{2A}$ | $\varphi_{2A}$ | $U_d$ | $\varphi_d$ |
|---|---|---|---|---|---|
| | | | | | |

D. Преобразование формы сигнала в частотно-зависимых цепях.

1. Переключить генератор в режим прямоугольных импульсов. Собрать RL-цепь или RC-цепь согласно указаниям преподавателя. Наблюдать изменение формы прямоугольных импульсов на выходе цепи. Зарисовать осциллограммы.

2. Представить результаты измерений преподавателю и после его проверки и одобрения выключить приборы и разобрать схему.

E. Самостоятельная работа.

1. По экспериментальным данным построить амплитудно-частотные и фазочастотные характеристики исследованных цепей, векторные диаграммы токов и напряжений.

2. По экспериментальным данным п. А-9 и п. А-10 для одной из частот в середине исследованного диапазона рассчитать сопротивление потерь катушки индуктивности RL и проводимость потерь конденсатора Gc. Для этого надо рассчитать комплексное входное сопротивление реактивного двухполюсника, включенного между точками 1A и 2A, по формуле

$$\underline{Z}_{вх} = \frac{\underline{U}_1 - \underline{U}_2}{\underline{I}} = \frac{\underline{U}_1 - \underline{U}_2}{\frac{\underline{U}_2}{R_{10}}} = R_{10}\left(\frac{U_1}{U_2}\cdot e^{j\varphi} - 1\right)$$

и найти активные составляющие $R_L$ = Rвх или $G_C$ = 1/Rвх. Найти добротность катушки

$$Q_L = \frac{X_L}{R_L}$$

и конденсатора $Q_C = X_C * G_C$.

3. Для RLC-цепи рассчитать и построить векторную диаграмму тока и напряжений для значений фазового угла $\varphi = +45^o$ и $\varphi = -45^o$.

4. По данным таблицы 4 построить векторную диаграмму токов и напряжений в цепи. Используя диаграмму, рассчитать значения токов в ветвях трехэлементного двухполюсника.

**Лабораторная работа № 3. Исследование последовательного колебательного контура**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 1.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

**Цель работы.** Исследование амплитудно-частотных характеристик последовательного колебательного контура, определение резонансной частоты $f_p$, полосы пропускания П, добротности $Q$ контуров.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

*Набор элементов и приборов, используемых в лабораторной работе*

Элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

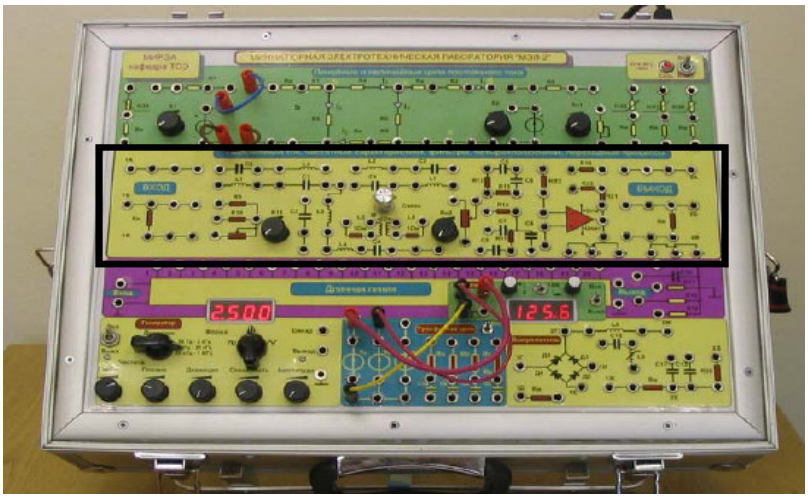


Рисунок 1. Панель МЭЛ для моделирования цепей переменного тока

**2 Теоретические сведения и методы численного расчета последовательного колебательного контура**

Резонансными называют цепи переменного тока, в которых при определенных условиях входной ток и входное напряжение совпадают по фазе, входное сопротивление цепи становится чисто активным, а ток в цепи или напряжения на элементах цепи становятся максимальными.

Схема последовательного колебательного контура показана на рисунке 2.

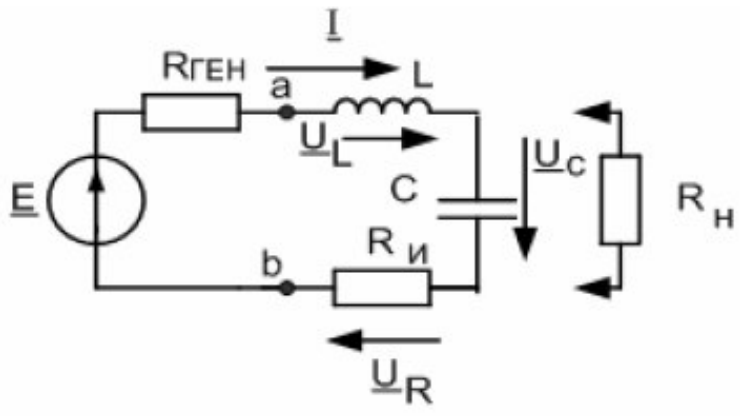



Рисунок 2. Схема последовательного контура

Контур «ab» подключен к источнику гармонического напряжения с внутренним сопротивлением $R_{ген}$.

Основные расчетные формулы для резонансной частоты $f_p$, характеристического сопротивления ρ, добротности Q, полосы пропускания П, нормированной амплитудно-частотной характеристики тока n(f), фазочастотной характеристики тока φ(f) следующие:

$$f_p = \frac{1}{2\pi\sqrt{LC}}; \qquad \rho = \sqrt{\frac{L}{C}}; \qquad Q = \frac{\rho}{R};$$

$$\Pi = \frac{f_p}{Q}; \qquad n(f) = \frac{1}{\sqrt{1 + Q^2\left(\frac{f}{f_p} - \frac{f_p}{f}\right)}};$$

$$\varphi(f) = \operatorname{arctg}\left[Q\left(\frac{f}{f_p} - \frac{f_p}{f}\right)\right].$$

Сопротивление контура

$$R = R_L + R_и,$$

где $R_L$ – сопротивление катушки; $R_и$ – измерительное сопротивление.

Комплексный ток на произвольной частоте можно вычислить по формуле $\underline{I} = \underline{I}_p \cdot n(f) \cdot e^{-\varphi(f)}$. Подключение сопротивления нагрузки к емкости контура снижает добротность.

Эквивалентная добротность контура с учетом нагрузки, измерительного сопротивления и внутреннего сопротивления источника сигнала вычисляется по формуле

$$Q_{экв} = \frac{\rho}{R_{ген} + R + \rho^2 / R_н}.$$

**3 Практическая часть**

Схема измерений собирается из элементов второй панели МЭЛ и показана на рисунке 3.

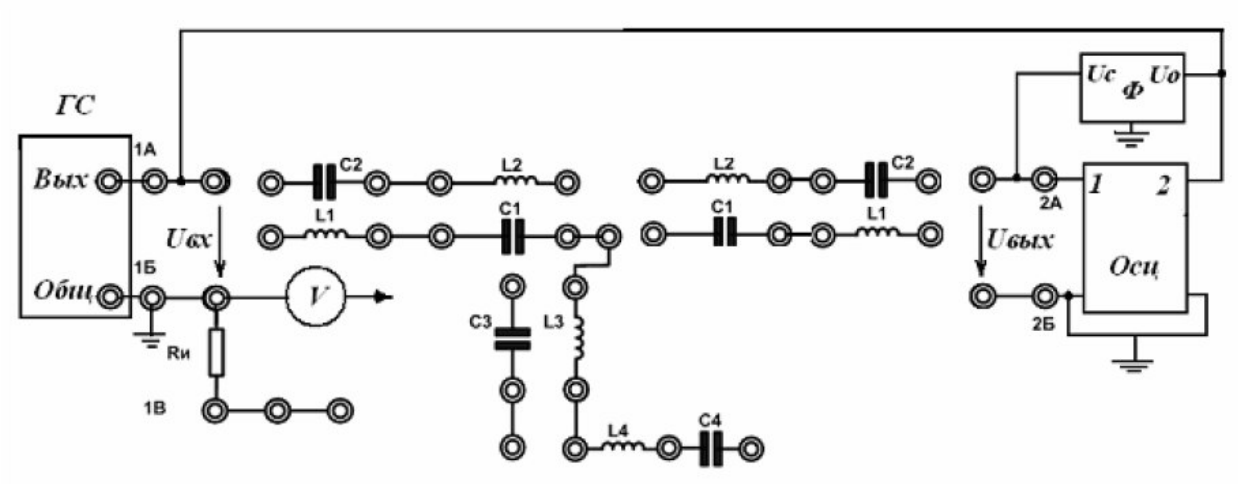



Рисунок 3. Схема моделирования контуров на МЭЛ

Схема содержит функциональный генератор сигналов, двухканальный осциллограф, электронный вольтметр.

Генератор сигналов установить в режим формирования гармонических сигналов, начальную частоту установить 1 кГц, выходное напряжение 100 мВ, девиацию частоты выключить.

1. Собрать схему последовательного контура (рисунок 4), включив в него для измерения тока измерительное сопротивление $R_и$ = 10 Ом.

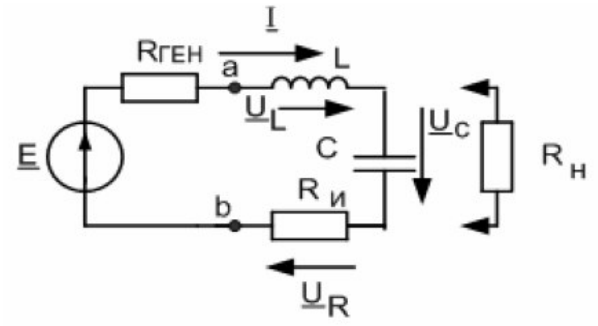



Рисунок 4. Схема последовательного контура

Значения индуктивности контура $L$ и емкости $C$ выбирают по заданию преподавателя из таблицы 1 и записывают их в протокол измерений.

Таблица 1

| № | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $L$ | $L_1$ | $L_1$ | $L_2$ | $L_2$ | $L_2$ | $2L_1$ | $2L_1$ | $L_2$ | $2L_1$ | $2L_2$ | $L_1$ | $L_1$ |
| $C$ | $C_1$ | $C_2$ | $C_1$ | $C_2$ | $0{,}5C_2$ | $0{,}5C_1$ | $C_2$ | $0{,}5C_1$ | $C_1$ | $C_1$ | $2C_1$ | $0{,}5C_1$ |

В стендах МЭЛ использованы следующие номиналы элементов: $L_1$ = 10 мГн, $L_2$ = 6,8 мГн, $C_1$ = 68 нФ, $C_2$ = 47 нФ. Для получения значений других значений из таблицы 1 элементы стенда надо соединять последовательно.

2. По значениям индуктивности и емкости рассчитать резонансную частоту контура.

3. Плавно изменяя частоту генератора, найти экспериментальное значение резонансной частоты и сравнить его с расчетным.

Резонансную частоту следует определять по нулевому сдвигу фаз между входным напряжением и током в контуре.

4. Изменяя частоту генератора в пределах от 1 до 20 кГц, провести измерение АЧХ-напряжения на измерительном сопротивлении электронным вольтметром и сдвига фаз между током и входным напряжением с помощью двухканального осциллографа или фазометра. Осциллограф включить в режим внутренней синхронизации. Напряжением $U_{вых}$ будет напряжение на измерительном сопротивлении.

Результаты записать в таблицу 2.

Таблица 2

| № | $f$, кГц | $U_{вых}$, мВ | $I$, мА | φ, град |
|---|---|---|---|---|
| | | | | |
| | | | | |

5. Определить полосу пропускания контура. Для этого на резонансной частоте, регулируя выходное напряжение генератора, установить стрелку вольтметра на отметку «0 дб». Затем, изменяя частоту генератора, найти нижнюю и верхнюю границы полосы пропускания по уровню «-3 дб». При этом ток в контуре $I_{гр}$ = 0,707$I_р$. Полоса пропускания П = fв – fн. Определить полосу пропускания и добротность по АЧХ и ФЧХ.

6. Подключить параллельно емкости сопротивление нагрузки $R_{14}$ = 10 кОм. Повторить измерения по п.п. 3–5 и заполнить таблицу, аналогичную 2.

7. Для заданной схемы контура рассчитать резонансную частоту, добротность, полосу пропускания, максимальный ток при резонансе с учетом внутреннего сопротивления генератора $R_{ген} \approx 10$ Ом в двух случаях: при выключенной и включенной нагрузке $R_{14}$.

8. Построить экспериментальные резонансные характеристики тока в контуре по таблицам 1 и 2. По графикам определить резонансные частоты, полосу пропускания, добротность и сравнить с расчетными значениями.

**Лабораторная работа № 4. Трехфазные электрические цепи**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 1.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

**Цель работы:** исследование режимов работы трехфазных цепей переменного тока при различных способах соединения симметричных и несимметричных нагрузок. Построение векторных диаграмм токов и напряжений в трехфазных цепях.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Описание набора элементов и приборов***

Пассивные элементы, используемые в данной лабораторной работе, размещены на пятой панели МЭЛ (рисунок 1).

Рисунок 1. Панель МЭЛ для моделирования трехфазных электрических цепей

## 2 Теоретические сведения и методы расчета трехфазных электрических цепей

Трехфазная система ЭДС вырабатывается в трехфазных синхронных генераторах и представляет три ЭДС с равной амплитудой и фазовым сдвигом $120^o$:

$$e_A = E_m \sin \omega t;\ e_B = E_m \sin(\omega t - 120^o);\ e_C = E_m \sin(\omega t + 120^o)\ .$$

На рисунке 2 показаны временные диаграммы трехфазных ЭДС. На рисунке 3 показана векторная диаграмма комплексных действующих ЭДС в трехфазной цепи.

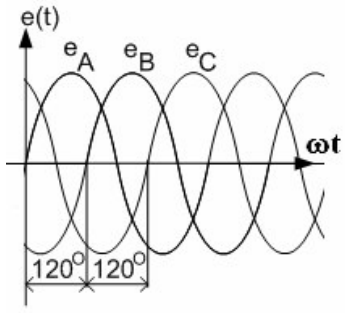



Рисунок 2. Временные диаграммы трехфазной системы ЭДС

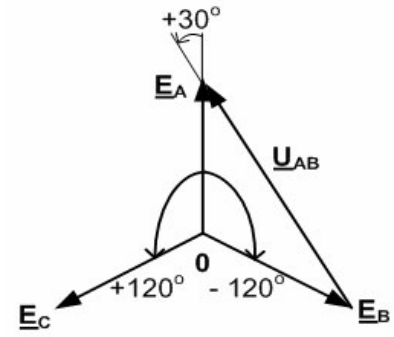



Рисунок 3. Векторная диаграмма

Применяют несколько способов соединения трехфазного источника ЭДС с нагрузками.

***Соединение звезда – звезда***

Соединение звезда – звезда показано на рисунке 4.

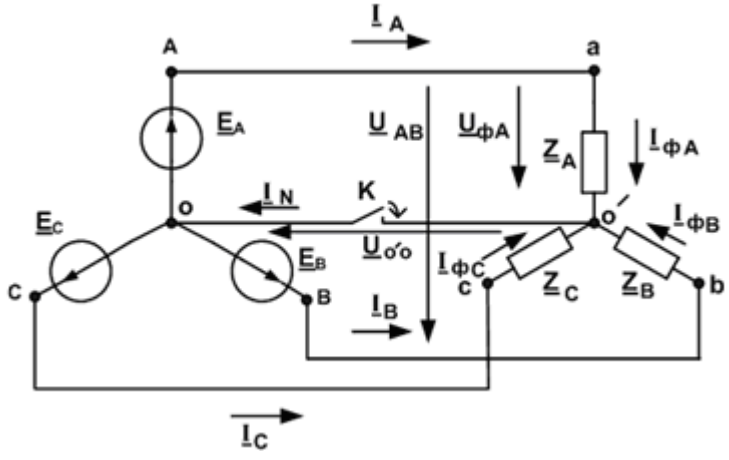



Рисунок 4. Соединение звезда – звезда

Трехфазные ЭДС $\underline{E}_A$, $\underline{E}_B$, $\underline{E}_C$ объединены в узле 0. Фазные нагрузки $\underline{Z}_A$, $\underline{Z}_B$, $\underline{Z}_C$ объединены в узле 0′. Провод, соединяющий узлы 00′, называют нейтральным проводом. В случае симметричной нагрузки $\underline{Z}_A = \underline{Z}_B = \underline{Z}_C$ применяют трехпроводное соединение звезда-звезда без нейтрального провода (ключ «К» разомкнут).

Токи в линейных проводах $\underline{I}_A$, $\underline{I}_B$, $\underline{I}_C$ в схеме рисунка 4 равны, соответственно, фазным токам в нагрузках $\underline{I}_{\phi A}$, $\underline{I}_{\phi B}$, $\underline{I}_{\phi C}$.

При симметричных нагрузках фазные напряжения на нагрузках равны фазным ЭДС, а фазные токи можно вычислить по формуле

$$\underline{I}_{\phi A} = \frac{\underline{E}_A}{\underline{Z}_A} = \frac{\underline{U}_{\phi A}}{\underline{Z}_A}.$$

В этом случае ток в нейтральном проводе $\underline{I}_N = 0$ и можно применять трехпроводную сеть.

При несимметричных нагрузках и включенном нейтральном проводе фазные токи не равны и возникает ток в нейтральном проводе $\underline{I}_N = \underline{I}_{\phi A} + \underline{I}_{\phi B} + \underline{I}_{\phi C}$.

Если нагрузки несимметричны, а нейтральный провод отсутствует, то между узлами 0′ и 0 возникает напряжение смещения нейтрали, которое рассчитывают по формуле

$$\underline{U}_{0'0} = \frac{\underline{E}_A\underline{Y}_A + \underline{E}_B\underline{Y}_B + \underline{E}_C\underline{Y}_C}{\underline{Y}_A + \underline{Y}_B + \underline{Y}_C},$$

где $\underline{Y}_A$,$\underline{Y}_B$,$\underline{Y}_C$ – комплексные проводимости фазных нагрузок. При этом токи в фазах нагрузки определяются по формулам вида

$$\underline{I}_{\phi A} = \frac{\underline{E}_A - \underline{U}_{0'0}}{\underline{Z}_A}.$$

Напряжение между линейными проводами называют линейным напряжением. В схеме рисунка 4 линейное напряжение

$$\underline{U}_{AB} = \sqrt{3}\underline{E}_A e^{+j30^0}.$$

***Соединение звезда – треугольник***

Соединение звезда-треугольник показано на рисунке 5.

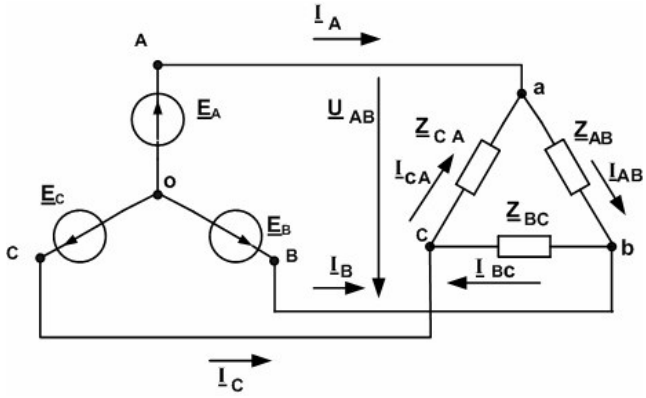



Рисунок 5. Соединение звезда – треугольник

К фазным нагрузкам приложены равные линейные напряжения $U_Л = \sqrt{3}E$.

Поэтому фазные нагрузки могут быть несимметричными. Фазные токи вычисляют по формуле вида

$$\underline{I}_{AB} = \frac{\underline{U}_{AB}}{\underline{Z}_{AB}}.$$

Линейные токи определяем по формулам вида $\underline{I}_A = \underline{I}_{AB} - \underline{I}_{CA}$.

***Мощность в трехфазной цепи***

При любом соединении и любой нагрузке комплексная мощность фазы нагрузки равна

$$\tilde{S} = \underline{U}_Ф \underline{I}_Ф^*.$$

Суммарная комплексная мощность трех фаз

$$\tilde{S}_\Sigma = \tilde{S}_A + \tilde{S}_B + \tilde{S}_C.$$

Отсюда можно получить выражения для полной активной мощности $P_\Sigma = P_A + P_B + P_C$ и полной реактивной мощности $Q_\Sigma = Q_A + Q_B + Q_C$.

При симметричной нагрузке $P_A = P_B = P_C = P_Ф$, полная активная мощность

$$P_\Sigma = 3U_Ф I_Ф \cos\varphi = \sqrt{3}U_Л I_Л \cos\varphi,$$

где φ – сдвиг фазы фазного тока относительно одноименного фазного напряжения.

***Лабораторное задание***

Схема измерений на МЭЛ показана на рисунке 6.

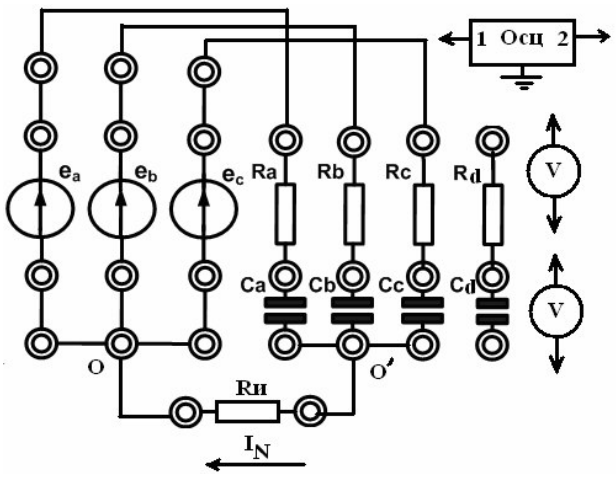


Рисунок 6. Схема измерений на МЭЛ

Трехфазный генератор $e_A$, $e_B$, $e_C$ формирует три напряжения с амплитудой около 4 В, частотой 50 Гц, сдвинутые по фазе на $120^o$.

Комплексные фазные нагрузки содержат резисторы Ra, Rb, Rc = 2 кОм и конденсаторы Ca, Cb, Cc = 2,2 мкФ. Набор этих элементов позволяет формировать симметричные и несимметричные нагрузки (активные, реактивные, комплексные). Дополнительное четвертое комплексное сопротивление $Z_{RdCd}$ служит для создания несимметричных нагрузок. Измерения производятся двухлучевым осциллографом и мультиметрами.

1. Собрать схему измерений рисунка 6 с нейтральным проводом. Для измерения тока в нейтральном проводе использовать измерительное сопротивление $R_и$ = 10 Ом.

2. Измерить осциллографом параметры трехфазных ЭДС, зарисовать на одном графике временные диаграммы напряжений $e_A$, $e_B$, $e_C$ с учетом фазового сдвига. Фазовый сдвиг измерить фазометром или по осциллографу. Измерить мультиметром действующие значения напряжений $e_A$, $e_B$, $e_C$ и записать результаты.

А. Исследование соединения звезда – звезда.

1. Подключить к трехфазному генератору соединенную звездой симметричную активную нагрузку из сопротивлений Ra, Rb, Rc. Мультиметрами провести измерения напряжений на фазных сопротивлениях, измерительном сопротивлении Ri в четырехпроводной системе. Результаты записать в таблицу 1.

Таблица 1

| Вид нагрузки | Кол. пров. | $U_{Ra}$ | $U_{Rb}$ | $U_{Rc}$ | $U_{Ca}$ | $U_{Cb}$ | $U_{Cc}$ | $U_{oo'}$ | $I_N$ |
|---|---|---|---|---|---|---|---|---|---|
| Сим. R | 4 | | | | | | | | |
| Сим. R | 3 | | | | | | | | |
| Несим. R | 4 | | | | | | | | |
| Несим. R | 3 | | | | | | | | |
| Сим. C | 4 | | | | | | | | |
| Сим. C | 3 | | | | | | | | |
| Несим. C | 4 | | | | | | | | |
| Несим. C | 3 | | | | | | | | |
| Сим. Z | 4 | | | | | | | | |
| Сим. Z | 3 | | | | | | | | |
| Несим. Z | 4 | | | | | | | | |
| Несим. Z | 3 | | | | | | | | |
| КЗ $\underline{Z}_A$ | 3 | | | | | | | | |
| ХХ $\underline{Z}_A$ | 3 | | | | | | | | |

2. Разомкнуть нейтральный провод. Мультиметрами провести измерения напряжений на фазных сопротивлениях и напряжение смещения нейтрали U00′ в трехпроводной системе. Результаты записать в таблицу 1.

3. Подключить по указанию преподавателя параллельно или последовательно к одному из фазных сопротивлений аналогичное по характеру дополнительное сопротивление.

4. Подключить к генератору соединенную звездой симметричную емкостную нагрузку Ca, Cb, Cc. Повторить измерения п.1–3. Результаты записать в таблицу 1.

5. Подключить к генератору соединенную звездой комплексную нагрузку $\underline{Z}_A$, $\underline{Z}_B$, $\underline{Z}_C$ . Повторить измерения п. 1–3. Результаты записать в таблицу 1.

B. Короткое замыкание фазы нагрузки.

1. Подключить к генератору соединенную звездой симметричную комплексную нагрузку $\underline{Z}_A$, $\underline{Z}_B$, $\underline{Z}_C$ без нейтрального провода**.** Закоротить перемычкой нагрузку $\underline{Z}_A$. Провести измерения и записать результаты в таблицу 1.

C. Обрыв линейного провода.

1. Подключить к генератору соединенную звездой симметричную комплексную нагрузку $Z_A$, $Z_B$, $Z_C$ без нейтрального провода. Отключить линейный провод фазы A. Провести измерения и записать результаты в таблицу 1.

D. Исследование соединения звезда – треугольник.

Схема измерений для соединения звезда – треугольник показана на рисунке 7. Токи в фазах нагрузки обозначены $\underline{I}_{ab}$, $\underline{I}_{bc}$, $\underline{I}_{ca}$. Для измерения линейных токов используется измерительное сопротивление Rи = 10 Ом.

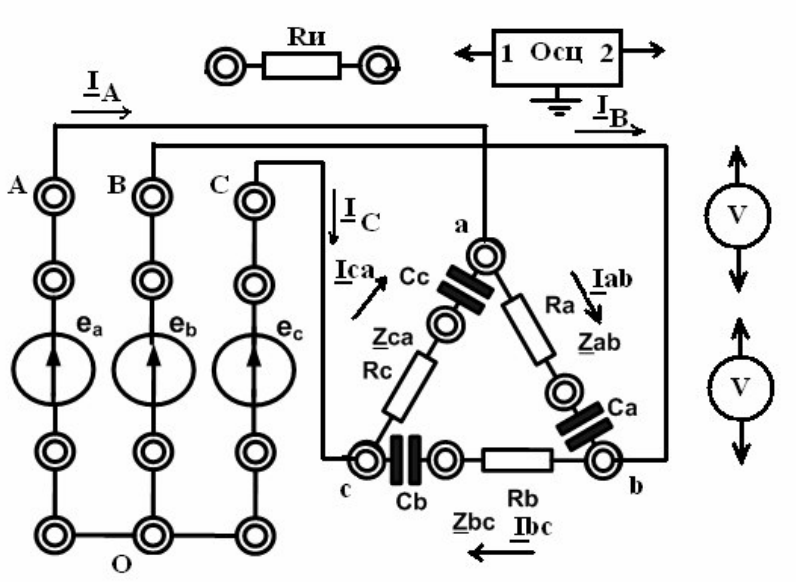



Рисунок 7. Соединение звезда-треугольник

1. Подключить к трехфазному генератору соединенную треугольником симметричную комплексную нагрузку Zab , Zbc , Zca . Мультиметром измерить линейные напряжения, линейные токи и напряжения на всех элементах нагрузки. Результаты измерений записать в таблицу 2.

Таблица 2

| Соединение звезда-треугольник | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Симметричная нагрузка | | | | | | | | | | | |
| $U_{ab}$ | $U_{bc}$ | $U_{ca}$ | $I_A$ | $I_B$ | $I_C$ | $U_{Ra}$ | $U_{Ca}$ | $U_{Rb}$ | $U_{Cb}$ | $U_{Rc}$ | $U_{Cc}$ |
| | | | | | | | | | | | |
| Несимметричная нагрузка | | | | | | | | | | | |
| $U_{ab}$ | $U_{bc}$ | $U_{ca}$ | $I_A$ | $I_B$ | $I_C$ | $U_{Ra}$ | $U_{Ca}$ | $U_{Rb}$ | $U_{Cb}$ | $U_{Rc}$ | $U_{Cc}$ |
| | | | | | | | | | | | |

2. По указанию преподавателя замкнуть один из элементов (R или C, но не оба) в одной из фазных нагрузок. Повторить измерения по п. 1. Результаты измерений записать в таблицу 2.

3. По результатам измерений таблицы 1 для каждого варианта нагрузок построить векторные диаграммы напряжений и токов в трехфазной цепи. Объяснить особенности режимов работы трехпроводной и четырехпроводной цепи при различных нагрузках.

4. Для соединения звезда – звезда рассчитать теоретически при несимметричных нагрузках в четырехпроводной системе ток нейтрали, а в трехпроводной системе – напряжение смещения нейтрали. Сравнить расчетные и экспериментальные результаты.

5. Для режима короткого замыкания нагрузки фазы А в трехпроводной системе рассчитать напряжения на элементах фазных нагрузок В и С.

6. Для режима обрыва линейного провода рассчитать напряжения и токи в фазных нагрузках В и С.

7. По результатам измерений таблицы 2 для соединения звезда – треугольник построить векторные диаграммы напряжений и токов. По диаграммам определить сдвиг фаз между линейными напряжениями и токами в нагрузках. Рассчитать активные и реактивные мощности в нагрузках. Определить комплексные значения токов в нагрузках Iab, Ibc, Ica. Вычислить по формулам линейные токи IA, IB, IC и сравнить с экспериментальными значениями этих токов.

**Лабораторная работа № 5. Исследование четырехполюсника**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 2.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

**Цель работы** – определение параметров линейного пассивного четырехполюсника по входным сопротивлениям в режимах холостого хода и короткого замыкания, а также выбор сопротивления нагрузки четырехполюсника из условия выделения в ней максимальной активной мощности.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Описание набора элементов и приборов***

Пассивные элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

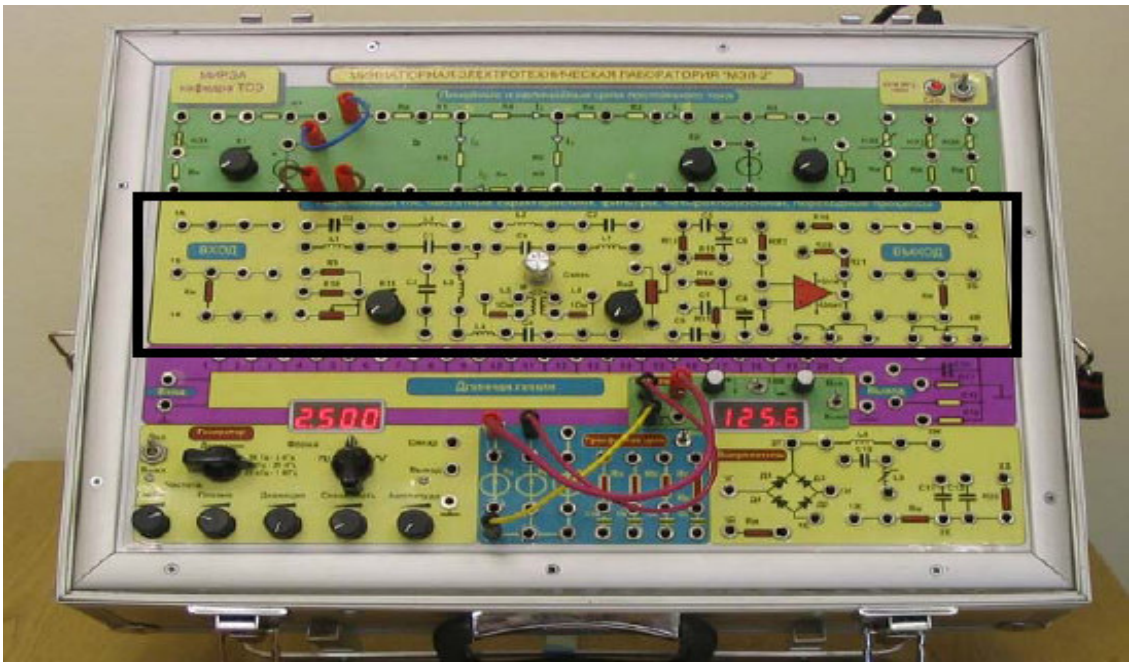



Рисунок 1. Панель МЭЛ для моделирования четырехполюсника

**2 Теоретические сведения и методы расчета линейных пассивных четырехполюсников**

Пассивный линейный четырехполюсник представляет собой элемент цепи, не содержащий источников энергии и имеющий два входных (первичных) (a-b) и два выходных (вторичных) (m-n) зажима (рисунок 2).

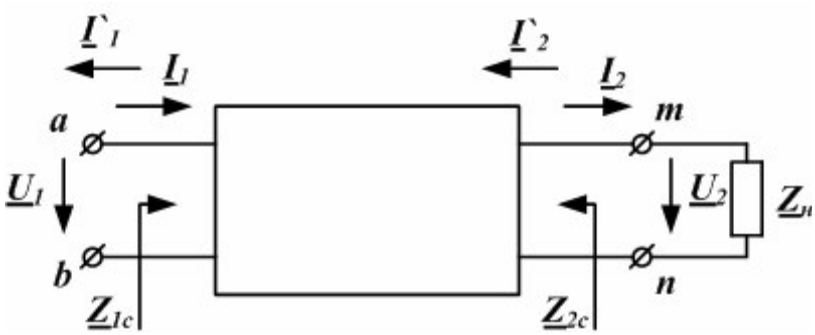



Рисунок 2. Схема четырехполюсника

Основные уравнения четырехполюсника могут быть записаны в шести различных формах, использующих параметры Y, Z, A, B, H, G.

Форма Y выражает токи $I_1$ и $I'_2$ через проводимости и напряжения:

$$\underline{I}_1 = \underline{Y}_{11} \cdot \underline{U}_1 + \underline{Y}_{12} \cdot \underline{U}_2 \ ;$$
$$\underline{I}'_2 = \underline{Y}_{21} \cdot \underline{U}_1 + \underline{Y}_{22} \cdot \underline{U}_2 \ .$$

Пассивные линейные четырехполюсники являются обратимыми. Для них выполняется теорема взаимности и взаимные проводимости $\underline{Y}_{12} = \underline{Y}_{21}$.

Форма A выражает входное напряжение $U_1$ и входной ток $I_1$ через выходное напряжение $U_2$ и выходной ток $I_2$. A-параметры применяются при анализе передачи энергии от входных зажимов к выходным зажимам. Для комплексных действующих значений уравнения четырехполюсника в форме A имеют вид

$$\underline{U}_1 = \underline{A}_{11}\underline{U}_2 + \underline{A}_{12}\underline{I}_2 \ ;$$
$$\underline{I}_1 = \underline{A}_{21}\underline{U}_2 + \underline{A}_{22}\underline{I}_2 \ . \qquad (1)$$

Коэффициенты уравнений четырехполюсника называют его параметрами. A-параметры четырехполюсника иногда именуют как коэффициенты A, B, C, D. Матрица A-параметров имеет вид

$$[A] = \begin{bmatrix} \underline{A}_{11} & \underline{A}_{12} \\ \underline{A}_{21} & \underline{A}_{22} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} .$$

Определитель А-параметров пассивного линейного четырехполюсника равен единице:

$$\left|A\right| = \underline{A}_{11} \cdot \underline{A}_{22} - \underline{A}_{12} \cdot \underline{A}_{21} = 1 \quad (2)$$

Это свойство надо использовать для проверки расчета А-параметров. Так как четыре параметра пассивного линейного четырехполюсника связаны последним уравнением, то независимыми являются только три параметра.

Физический смысл и непосредственное определение А-параметров:

$\underline{A}_{11} = \dfrac{\underline{U}_1}{\underline{U}_2}$ при $\underline{I}_2 = 0$ (режим холостого хода на выходе – ХХ2) – коэффициент трансформации по напряжению;

$\underline{A}_{22} = \dfrac{\underline{I}_1}{\underline{I}_2}$ при $\underline{U}_2 = 0$ (короткое замыкание на выходе – К32) – коэффициент трансформации тока;

$\underline{A}_{12} = \dfrac{\underline{U}_1}{\underline{I}_2}$ при $\underline{U}_2 = 0$ – величина, обратная передаточной проводимости при К32;

$\underline{A}_{21} = \dfrac{\underline{I}_1}{\underline{U}_2}$ при $\underline{I}_2 = 0$ – величина, обратная передаточному сопротивлению при ХХ2.

В симметричном четырехполюснике (токи и напряжения во внешней цепи не меняются при перемене местами первичных и вторичных зажимов) выполняется равенство $\underline{A}_{11} = \underline{A}_{22}$. Поэтому в симметричном пассивном линейном четырехполюснике два независимых параметра.

***Входное сопротивление четырехполюсника***

Входное сопротивление четырехполюсника со стороны первичных зажимов находят по формуле

$$\underline{Z}_{1вх} = \frac{\underline{U}_1}{\underline{I}_1} = \frac{\underline{A}_{11}\underline{U}_2 + \underline{A}_{12}\underline{I}_2}{\underline{A}_{21}\underline{U}_2 + \underline{A}_{22}\underline{I}_2} = \frac{\underline{A}_{11}\dfrac{\underline{U}_2}{\underline{I}_2} + \underline{A}_{12}}{\underline{A}_{21}\dfrac{\underline{U}_2}{\underline{I}_2} + \underline{A}_{22}} = \frac{\underline{A}_{11}\underline{Z}_2 + \underline{A}_{12}}{\underline{A}_{21}\underline{Z}_2 + \underline{A}_{22}}.$$

Входное сопротивление со стороны выходных зажимов находят по аналогичной формуле

$$\underline{Z}_{2вх} = \frac{\underline{U}_2}{\underline{I}_2} = \frac{\underline{A}_{22}\underline{U}_1 + \underline{A}_{12}\underline{I}_1}{\underline{A}_{21}\underline{U}_1 + \underline{A}_{11}\underline{I}_1} = \frac{\underline{A}_{22}\underline{Z}_1 + \underline{A}_{12}}{\underline{A}_{21}\underline{Z}_1 + \underline{A}_{11}}.$$

Из формул (1) и (2) следует, что четырехполюсник преобразует (трансформирует) сопротивление нагрузки.

Частными случаями входных сопротивлений являются сопротивления холостого хода:

$\underline{Z}_{1вх} = \underline{Z}_{1x} = \dfrac{\underline{A}_{11}}{\underline{A}_{21}}$ при $\underline{Z}_2 = \infty$ и $\underline{Z}_{2вх} = \underline{Z}_{2x} = \dfrac{\underline{A}_{22}}{\underline{A}_{21}}$ при $\underline{Z}_1 = \infty$;

сопротивления короткого замыкания:

$\underline{Z}_{1вх} = \underline{Z}_{1k} = \dfrac{\underline{A}_{12}}{\underline{A}_{22}}$ при $\underline{Z}_2 = 0$ и $\underline{Z}_{2вх} = \underline{Z}_{2к} = \dfrac{\underline{A}_{12}}{\underline{A}_{11}}$ при $\underline{Z}_1 = 0$.

А-параметры определяются по входным сопротивлениям, полученным опытным путем в режимах холостого хода (х.х.) и короткого замыкания (к.з.) по формулам

$$\underline{A}_{11}=\sqrt{\frac{\underline{Z}_{1X}}{\underline{Z}_{2X}-\underline{Z}_{2K}}},\ \underline{A}_{12}=\underline{A}_{11}\underline{Z}_{2K},\ \underline{A}_{21}=\frac{\underline{A}_{11}}{\underline{Z}_{1X}},\ \underline{A}_{22}=\underline{A}_{11}\frac{\underline{Z}_{2X}}{\underline{Z}_{1X}}.$$

***Выбор нагрузки из условия выделения в ней максимальной мощности $P_{макс}$***

Из теории известно, что при питании четырехполюсника от источника ЭДС для того, чтобы в нагрузке $Z_H = R_H + jX_H$ выделилась максимально возможная активная мощность $P_2$, необходимо, чтобы сопротивление нагрузки было комплексно сопряженным с входным сопротивлением четырехполюсника со стороны зажимов m – n при короткозамкнутых зажимах a – b.

Если $\underline{Z}_{BXmn} = R_{BX} + jX_{BX}$, то должно иметь место

$$\underline{Z}_H = R_H + jX_H = R_{BX} - jX_{BX} = \underline{Z}^*_{2K}.$$

Таким образом, $R_H = R_{BX}$ и $X_H = -X_{BX}$. При этом в нагрузке выделяется $P_{\max} = U_2^{\,2}/4R_H$, где $U_2$ – напряжение холостого хода на зажимах m-n.

***Схемы замещения четырехполюсника***

Если на некоторой фиксированной частоте определены А-параметры четырехполюсника, в расчетах и экспериментах этот четырехполюсник можно представить схемой замещения, которая имеет ту же матрицу А-параметров. Применяют две схемы замещения: Т-образная (рисунок 3) и П-образная (рисунок 4).

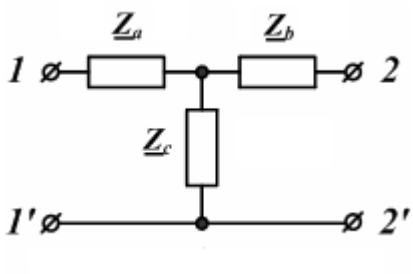



Рисунок 3. Т-образная схема замещения

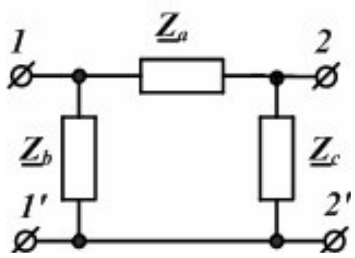



Рисунок 4. П-образная схема замещения

Расчет элементов Т-образной схемы замещения через А-параметры четырехполюсника проводят по формулам:

$$\underline{Z}_a=\frac{\underline{A}_{11}-1}{\underline{A}_{21}};\ \underline{Z}_b=\frac{\underline{A}_{22}-1}{\underline{A}_{21}};\ \underline{Z}_c=\frac{1}{\underline{A}_{21}}.$$

Расчет элементов П-образной схемы замещения проводят по формулам:

$$\underline{Z}_a=\underline{A}_{12};\ \underline{Z}_b=\frac{\underline{A}_{12}}{\underline{A}_{22}-1};\ \underline{Z}_c=\frac{\underline{A}_{12}}{\underline{A}_{11}-1}.$$

***Характеристические параметры четырехполюсника***

Характеристическими параметрами четырехполюсника называют два характеристических сопротивления:

$$\underline{Z}_{1C} = \sqrt{\frac{\underline{A}_{11} \cdot \underline{A}_{12}}{\underline{A}_{21} \cdot \underline{A}_{22}}} = \sqrt{\underline{Z}_{1K} \cdot \underline{Z}_{1X}}\,;\ \underline{Z}_{2C} = \sqrt{\frac{\underline{A}_{22} \cdot \underline{A}_{12}}{\underline{A}_{21} \cdot \underline{A}_{11}}} = \sqrt{\underline{Z}_{2K} \cdot \underline{Z}_{2X}}$$

и характеристическую постоянную передачи (меру передачи)

$$\underline{g} = \ln(\sqrt{\underline{A}_{11} \cdot \underline{A}_{22}} + \sqrt{\underline{A}_{12} \cdot \underline{A}_{21}}) = a + jb\,.$$

Характеристические сопротивления обладают таким свойством, что если к вторичным зажимам подключить в качестве нагрузки $\underline{Z}_{2C}$, то входное сопротивление со стороны первичных зажимов будет равно $\underline{Z}_{1C}$. И наоборот, если к первичным зажимам подключить $\underline{Z}_{1C}$, то входное сопротивление со стороны выходных зажимов будет $\underline{Z}_{2C}$.

Четырехполюсник, нагруженный на характеристическое сопротивление, называют согласованным с нагрузкой. Согласованный режим работы является весьма важным и часто используется на практике. Нагрузка, равная характеристическому сопротивлению, также называется согласованной.

В согласованном режиме напряжения на входе и выходе четырехполюсника выражаются через характеристические параметры по формуле

$$\underline{U}_1 = \sqrt{\frac{\underline{Z}_{1C}}{\underline{Z}_{2C}}} \cdot \underline{U}_2 \cdot e^{\underline{g}} = \sqrt{\frac{\underline{Z}_{1C}}{\underline{Z}_{2C}}} \cdot \underline{U}_2 \cdot e^{a} \cdot e^{jb}\,,$$

где a – характеристическое затухание четырехполюсника, b – характеристическая фаза.

Комплексная передаточная функция четырехполюсника выражается через А-параметры и сопротивление нагрузки следующей формулой:

$$\underline{K}_U(j\omega) = \frac{\underline{U}_2}{\underline{U}_1} = \frac{\underline{Z}_2}{\underline{A}_{11}\underline{Z}_2 + \underline{A}_{12}} = K_U(\omega) \cdot e^{j\varphi(\omega)}\,.$$

***Лабораторное задание***

Схема измерений представлена на рисунке 5 и содержит генератор синусоидальных сигналов, электронный вольтметр, фазометр и осциллограф. Измерительные сопротивления Rи на второй панели МЭЛ равны 10 Ом.

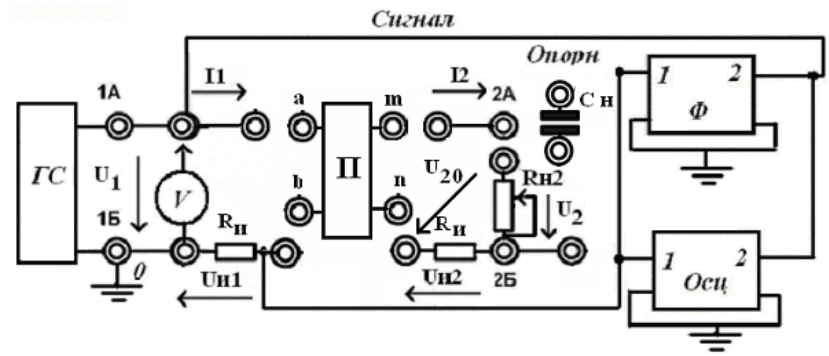



Рисунок 5. Схема реального моделирования

Сам четырехполюсник с зажимами ab – mn собирается из пассивных элементов второй панели МЭЛ по схеме регулярного четырехполюсника с объединенными зажимами b – n, заданной преподавателем. Пример возможной схемы показан на рисунке 6.

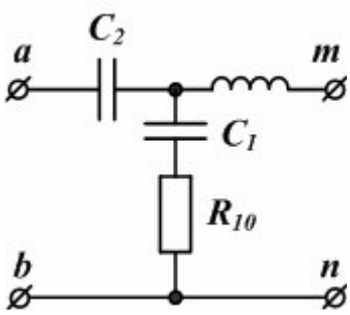



Рисунок 6. Пример схемы четырехполюсника

В ветви с сопротивлением $R_{10}$ обязательно включите индуктивность или емкость. Четырехполюсник должен содержать хотя бы одну индуктивность для получения резонансного режима.

1. Для реальной модели собрать схему четырехполюсника, заданную преподавателем, и схему измерений (см. рисунок 5). Соединить зажим *a* четырехполюсника с клеммой 1А, зажим *b* с клеммой 1Б, зажим *m* соединить с клеммой 2А, зажим *n* соединить с клеммой 2Б через измерительный резистор $R_И$. Подключить к клеммам 2А–2Б конденсатор Сн (один из конденсаторов $C_1$–$C_4$, не использованный в схеме четырехполюсника). Включить вольтметр к клеммам 2А–2Б. Установить на выходе ГС напряжение 1В.

Изменяя частоту ГС, найти максимальное напряжение на конденсаторе, соответствующее резонансному режиму выходной цепи. Записать значение резонансной частоты $f_p$.

На резонансной частоте измерить и записать напряжение на конденсаторе нагрузки $U_2$ и напряжение на измерительном сопротивлении $U_{И2}$. Рассчитать реактивное сопротивление конденсатора нагрузки $X_{СН}$.

2. Провести опыт прямого холостого хода. В схеме (см. рисунок 5) с фазометром, осциллографом и вольтметром отключить конденсатор от выходных зажимов четырехполюсника, оставив разомкнутыми клеммы 2А–2Б. Между клеммами 1Б и b включить измерительный резистор $R_И$. Вольтметром измерить напряжения $U_1$ и $U_{и1}$. Фазометром или осциллографом измерить угол сдвига фаз между током $I_1$ и напряжением $U_1$ в режиме холостого хода.

Результаты измерений записать в таблицу 1 (прямой опыт ХХ).

Таблица 1

| Наименование опыта | $U_1$ | $U_{и1}$ | φ | $I_1$ | $Z$ |
|---|---|---|---|---|---|
| Прямой опыт ХХ | | | | | $\underline{Z}_{1X}$= |
| Прямой опыт КЗ | | | | | $\underline{Z}_{1К}$= |
| Обратный опыт ХХ | | | | | $\underline{Z}_{2X}$= |
| Обратный опыт КЗ | | | | | $\underline{Z}_{2К}$= |

По этим данным можно подсчитать $Z_{1X}$:

$$I_1 = \frac{U_{И1}}{10};\ Z_{вх} = U_1 / I_1;\ R_{1X} = Z_{вх} \cdot \cos\varphi_{вх} - R_и;\ X_{1x} = Z_{вх} \cdot \sin\varphi_{вх};$$

$$Z_{1X} = \sqrt{R_{1X}^2 + X_{1X}^2},\ \varphi_{1X} = \operatorname{arc\,tg}\frac{X_{1X}}{R_{1X}},\ \underline{Z}_{1X} = Z_{1X} \cdot e^{j\varphi_{1X}}.$$

3. Замкнуть клеммы m–n. Повторить измерения по п. 2. Записать результаты в таблицу 1 (прямой опыт КЗ). Рассчитать $\underline{Z}_{1К}$.

4. В схеме (см. рисунок 5) поменять местами зажимы a–b и m–n четырехполюсника. Разомкнуть клеммы a-b. Повторить измерения по п. 2. Записать результаты в таблицу 1 (опыт обратного холостого хода ХХ). Рассчитать $\underline{Z}_{2X}$.

5. Замкнуть клеммы a-b. Повторить измерения по п. 2. Записать результаты в таблицу 1 (обратный опыт КЗ). Рассчитать $\underline{Z}_{2К}$.

6. По данным опытов 2-5 проверить выполнение соотношения

$$\underline{Z}_{1к}/\underline{Z}_{1х} = \underline{Z}_{2к}/\underline{Z}_{2х}.$$

7. В схеме (см. рисунок 5) соединить зажимы четырехполюсника a-b с клеммами 1А-1Б, зажим m соединить с клеммой 2А, зажим n соединить с клеммой 2Б через измерительный резистор $R_И$. Подключить к клеммам 2А-2Б нагрузку $R_{н2}$ и последовательно с ней конденсатор нагрузки $C_Н$ из п. 1, соединив его с клеммой 2А. Ток $I_2$ в активной нагрузке $R_{н2}$ совпадает по фазе с напряжением $U_2$ и $\varphi_2 = 0$. Изменяя значение нагрузки от нуля до максимального значения, измерить 6–7 значений $U_{20}$, $U_{И2}$. Напряжение $U_2 = U_{20} – U_{И2}$.

Записать результаты в таблицу 2.

Таблица 2

| | $R_{н2}$ | $U_{20}$ | $U_{и2}$ | $U_2$ | $I_2$ | $P_2$ |
|---|---|---|---|---|---|---|
| Сн вкл | 0 | | | | | |
| | | | | | | |
| | мах | | | | | |
| Сн выкл | $R_{н2опт}$ | | | | | |

Вычислить значения активной мощности в нагрузке по формуле $P_2 = U_2*I_2*\cos\varphi_2$.

8. Установить оптимальное значение активной нагрузки $R_{н2опт}$, соответствующее максимальной активной мощности. Изменяя в небольших пределах частоту генератора, убедиться в снижении напряжения $U_2$ при отклонении частоты от резонансной и нарушении согласования реактивных сопротивлений.

9. Установить оптимальное значение нагрузки $R_{н2опт}$, соответствующее максимальной активной мощности, и резонансную частоту. Закоротить конденсатор. Измерить $U_2$, $U_{И2}$ и вычислить мощность в несогласованной нагрузке.

10. По данным опытов 1 и 7 записать оптимальное сопротивление комплексной нагрузки $Z_{нопт} = R_{н2опт} - jX_{Сн}$.

11. В схеме (см. рисунок 5) входы 1 фазометра и осциллографа подключить к клемме 1А. Входы 2 подключить к клемме 2А. Измерительные сопротивления $R_и$ закоротить.

К выходу четырехполюсника m-n подключить нагрузку $R_{н2}$ без конденсатора. Изменяя сопротивление нагрузки от нуля до максимального значения, измерить напряжения на $U_1$, $U_2$ и разность фаз φ. Провести измерения также для режима холостого хода, отключив сопротивление нагрузки. Вычислить комплексную передаточную функцию по напряжению

$$\underline{K}_{21} = \frac{U_2}{U_1}\exp(j\varphi).$$

Результаты записать в таблицу 3.

Таблица 3

| $R_{н2}$ | $U_1$ | $U_2$ | φ | $K_{21}$ |
|---|---|---|---|---|
| 0 | | | | |
| | | | | |
| мах | | | | |
| Режим ХХ | | | | |

**Лабораторная работа № 6. Исследование переходных процессов в цепях с сосредоточенными параметрами**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 2.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

***Цель работы.*** В работе исследуются переходные процессы в цепях первого порядка R, L и R, C , а также в цепи второго порядка R, L, C при апериодическом и колебательном характерах процесса.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Описание набора элементов и приборов***

Пассивные элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

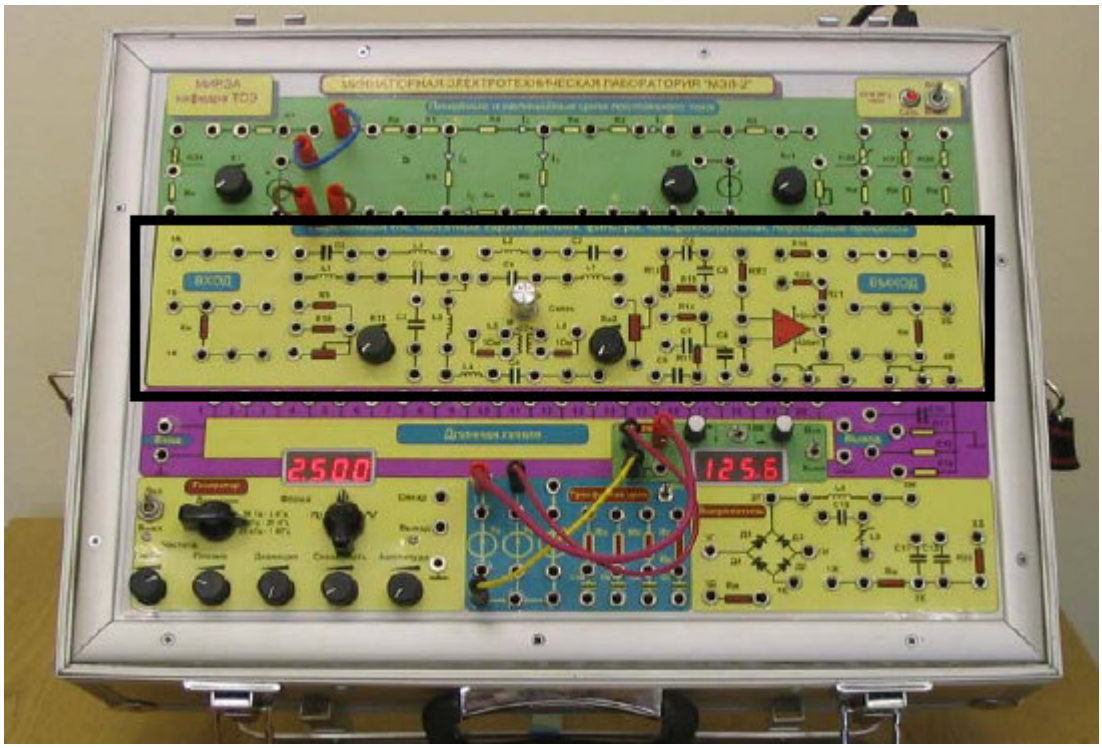


Рисунок 1. Панель МЭЛ для моделирования переходных процессов
в цепях с сосредоточенными параметрами

**2 Теоретические сведения и методы расчета переходных процессов в цепях с сосредоточенными параметрами**

Переходным процессом называется неустановившийся, нестационарный процесс, возникший при переходе из одного режима работы к другому. Всякие изменения и переключения в схеме называют коммутацией. В схеме рисунка 2 в момент t = 0 происходит коммутация (в данном случае замыкание ключа).

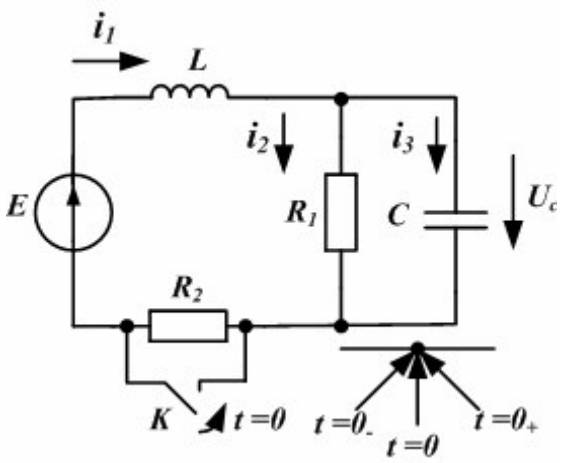



Рисунок 2. Схема цепи с коммутирующим ключом К

Режим работы цепи изменяется и возникает переходный процесс.

Считается, что коммутация происходит мгновенно в момент времени t = 0. Момент времени, предшествующий коммутации, обозначен t = 0_. Момент времени, следующий сразу после коммутации,

обозначен $t = 0_+$. Примем следующие значения параметров цепи: E = 120 В, L = 10 мГн, C = 68 нФ, $R_1 = R_2 = 1$ кОм.

До коммутации в момент $t = 0_-$ ток в индуктивности

$$i_1(0_-) = \frac{E}{R_1 + R_2}.$$

В индуктивности накоплена магнитная энергия

$$W_M(0_-) = \frac{L \cdot i_1^2(0_-)}{2}.$$

Энергия не может измениться мгновенно, так как мощность всегда ограничена

$$P(t) = \frac{dW}{dt} \neq \infty.$$

*Первый закон коммутации.* Ток в индуктивности до коммутации равен току в индуктивности в начальный момент после коммутации:

$$i_L(0_-) = i_L(0_+).$$

До коммутации в момент $t = 0_-$ напряжение на емкости

$$u_C(0_-) = \frac{E \cdot R_1}{R_1 + R_2}.$$

На емкости накоплена электрическая энергия

$$W_Э(0_-) = \frac{C \cdot u^2{}_C(0_-)}{2}.$$

Электрическая энергия также не может изменяться мгновенно.

*Второй закон коммутации.* Напряжение на емкости до коммутации равно напряжению на емкости в начальный момент после коммутации:

$$u_C(0_-) = u_C(0_+).$$

Расчет переходных процессов основан на использовании первого и второго закона коммутации.

Переходные процессы в линейных электрических цепях описываются линейными дифференциальными уравнениями. Для цепи, показанной на рисунке 1, систему дифференциальных уравнений составим по законам Кирхгофа:

$$i_1 = i_2 + i_3;$$

(1)

$$i_2 = \frac{u_C}{R_1};$$

(2)

$$i_3 = C\frac{du_C}{dt};$$

(3)

$$L\frac{di_1}{dt}+u_C=E\,.$$

(4)

Используя уравнения (1)–(3), преобразуем (4) к виду

$$\frac{d^2u_C}{dt^2}+\frac{1}{CR_1}\cdot\frac{du_C}{dt}+\frac{1}{LC}u_C=\frac{E}{LC}\,.$$

(5)

Получили линейное, однородное дифференциальное уравнение второго порядка. Расчет переходных процессов в линейной электрической цепи можно выполнить несколькими методами.

***Классический метод расчета переходных процессов***

В классическом методе переходное напряжение или ток ищут как сумму свободной и принужденной составляющей. Принужденную составляющую находят расчетом послекоммутационной цепи в установившемся принужденном режиме, когда после коммутации прошло много времени. Свободную составляющую ищут как общее решение однородного дифференциального уравнения при нулевом внешнем воздействии в виде

$$u_{Cсв}(t)=A_1\cdot e^{p_1t}+A_2\cdot e^{p_2t}+\ldots\,,$$

(6)

где $p_1$, $p_2$, … – корни характеристического уравнения, $A_1$, $A_2$, … – неизвестные постоянные интегрирования. Число корней характеристического уравнения и число неизвестных постоянных интегрирования равно порядку цепи, который определяется числом независимых накопительных реактивных элементов.

Для линейных цепей первого порядка характеристическое уравнение имеет один корень и свободная составляющая переходного процесса выражается одной экспоненциальной функцией из (6):

$$u_{Cсв}(t)=A_1\cdot e^{p_1t}\,.$$

(7)

Постоянной времени цепи первого порядка называют

$$\tau=\frac{1}{|p_1|}.$$

При этом

$$u_{Cсв}(t)=A_1\cdot e^{-\frac{t}{\tau}}.$$

Для линейных цепей второго порядка, которым соответствуют дифференциальные уравнения вида (5), характеристическое уравнение имеет следующий вид:

$$Z(p)=p^2+2\delta p+\omega^2{}_0=0\,.$$

(8)

Если $\delta > \omega_0$, корни характеристического уравнения отрицательные и разные. Переходный процесс описывается двумя затухающими экспонентами и называется апериодическим.

Если $\delta < \omega_0$, корни характеристического уравнения будут комплексносопряженными с отрицательной действительной частью. Переходный процесс имеет вид затухающих колебаний и называется колебательным.

***Электрические цепи при импульсных воздействиях***

На рисунке 3 показана форма импульсного сигнала с амплитудой E.

На рисунке 4 показана эквивалентная схема подключения цепи с сопротивлением и емкостью (RC-цепи) к генератору импульсного сигнала.

В момент t = 0 ключ K подключает RC-цепь к источнику напряжения E = 1 В с внутренним сопротивлением $R_{вн}$, и емкость заряжается. В момент t = T/2 ключ K закорачивает RC-цепь, и емкость разряжается. В момент t = T RC-цепь снова подключается к источнику напряжения.

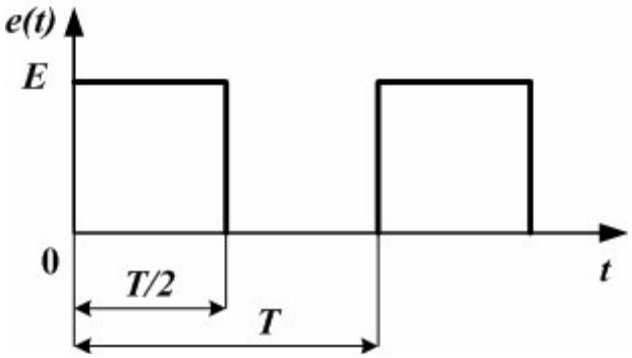



Рисунок 3. Форма импульсного сигнала

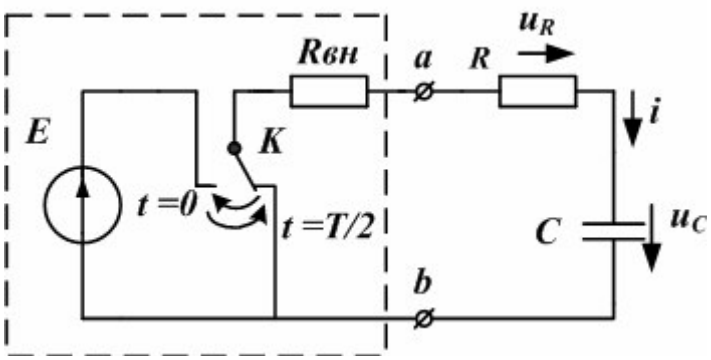



Рисунок 4. Эквивалентная схема подключения RC-цепи к генератору

Переходные процессы в цепях первого и второго порядка проще рассчитывать для напряжения на емкости или для тока в индуктивности.

***Расчет заряда емкости классическим методом***

Первый интервал времени: $0 < t \le \frac{T}{2}$.

1. Расчет режима до коммутации: $u_C(0_-) = u_C(0_+) = 0$.
2. Расчет принужденного режима (считаем, что включенное напряжение E сохраняется бесконечно долго). При этом емкость должна зарядится до величины E и, следовательно, $u_{Спр} = E$.
3. Дифференциальное уравнение цепи на первом интервале:

$$i \cdot (R + R_{вн}) + u_C = (R + R_{вн})C \cdot \frac{du_C}{dt} + u_C = E \,. \quad (9)$$

4. Характеристическое уравнение:

$$(R + R_{вн})C \cdot p + 1 = 0, \; p = -\frac{1}{(R + R_{вн})C} \,.$$

5. Свободная составляющая:

$$u_{Ссв}(t) = A \cdot e^{pt}.$$

Находим

$$A = u_{Ссв}(0_+) = u_C(0_+) - u_{Спр} = -E.$$

Следовательно,

$$u_{Ссв}(t) = -E \cdot e^{-\frac{t}{(R+R_{вн})C}}.$$

6. Полное напряжение на емкости:

$$u_C(t) = u_{Спр} + u_{Ссв}(t) = E(1 - e^{-\frac{t}{(R+R_{вн})C}}). \quad (10)$$

7. Полное напряжение на сопротивлении:

$$u_R(t) = RC \cdot \frac{du_C}{dt} = \frac{E \cdot R}{R + R_{вн}} \cdot e^{-\frac{t}{(R+R_{вн})C}}. \quad (11)$$

***Расчет разряда емкости***

Второй интервал: $\frac{T}{2} < t \le T$.

1. В момент $t = \frac{T}{2}$

и

$$c(\frac{T}{2}-) = u_C(\frac{T}{2}+) = E(1 - e^{-\frac{T/2}{(R+R_{вн})C}}).$$

2. В принужденном режиме $u_{Спр} = 0$ (считаем, что ключ закорачивает емкость бесконечно долго и емкость полностью разряжается).

3. Дифференциальное уравнение цепи на втором интервале:

$$(R + R_{вн})C \cdot \frac{du_C}{dt} + u_C = 0.$$

4. Характеристическое уравнение в данной схеме не изменяется:

$$(R + R_{вн})C \cdot p + 1 = 0,\ p = -\frac{1}{(R + R_{вн})C}.$$

5. Свободная составляющая напряжения на емкости:

$$u_{Ссв}(t) = B \cdot e^{pt} = u_{Ссв}(\frac{T}{2}+) \cdot e^{-\frac{t-T/2}{(R+R_{вн})C}} = E(1 - e^{-\frac{T/2}{(R+R_{вн})C}}) \cdot e^{-\frac{t-T/2}{(R+R_{вн})C}}.$$

6. Полное напряжение на емкости на втором интервале равно свободной составляющей, так как принужденная составляющая равна нулю:

$$u_C(t) = E(1 - e^{-\frac{T/2}{(R+R_{вн})C}}) \cdot e^{-\frac{t-T/2}{(R+R_{вн})C}}. \quad (12)$$

7. Напряжение на сопротивлении:

$$u_R(t) = RC \cdot \frac{du_C}{dt} = -\frac{E \cdot R}{R + R_{вн}} \cdot (1 - e^{-\frac{T/2}{(R+R_{вн})C}}) \cdot e^{-\frac{t-T/2}{(R+R_{вн})C}} . \quad (13)$$

*Переходный процесс разряда конденсатора при последовательном соединении индуктивности, конденсатора и активного сопротивления (цепь второго порядка)*

Дифференциальному уравнению $u_c + iR + L\frac{di}{dt} = 0$ электрической цепи второго порядка (рисунок 5) соответствует характеристическое уравнение

$$p^2 + \frac{R}{L}p + \frac{1}{LC} = 0 . \quad (14)$$

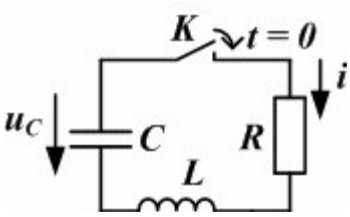



Рисунок 5. Разряд емкости в RLC-цепи

Характер переходного процесса зависит от значения корней уравнения (14):

$$p_{1,2} = -\frac{R}{2L} \pm \sqrt{\frac{R^2}{4L^2} - \frac{1}{LC}} .$$

При этом возможны три случая.

1. При $\frac{R^2}{4L^2} > \frac{1}{LC}$, т.е. при добротности контура $Q = \frac{\sqrt{L/C}}{R} < 0.5$, разряд в цепи имеет апериодический характер:

$$u_c = A_1 e^{p_1 t} + A_2 e^{p_2 t};$$
$$i = C(p_1 A_1 e^{p_1 t} + p_2 A_2 e^{p_2 t}) .$$

2. При $\frac{R^2}{4L^2} = \frac{1}{LC}$, т.е. при добротности контура Q = 0,5, в цепи имеет место предельный случай апериодического разряда конденсатора.

Сопротивление цепи $R = 2\sqrt{\frac{L}{C}}$ называется критическим, при этом

$$u_c = (A_1 + A_2 t)e^{pt} .$$

3. При $\frac{R^2}{4L^2} < \frac{1}{LC}$, т.е. при добротности контура Q>0.5, корни характеристического уравнения комплексно-сопряженные:

$$P_{1,2} = -\delta \pm j\omega_c,$$

где $\delta = \frac{R}{2L}$ – характеризует затухание процесса; $\omega_c = \sqrt{\frac{1}{LC} - \frac{R^2}{4L^2}}$ – угловая частота затухающих свободных колебаний.

Разряд конденсатора имеет колебательный характер с затухающей амплитудой:

$$u_c = Ae^{-\delta t}\sin(\omega_c t + v).$$

Для определения $\omega_c$ по изображению $u_c$ на экране осциллографа следует измерить $T_c$ – период колебаний, затем вычислить

$$\omega_c = \frac{2\pi}{T_c}.$$

Коэффициент затухания δ можно вычислить из отношения напряжений, взятых в моменты $t_i$ и $t_1 + T_c$ (рисунок 6).

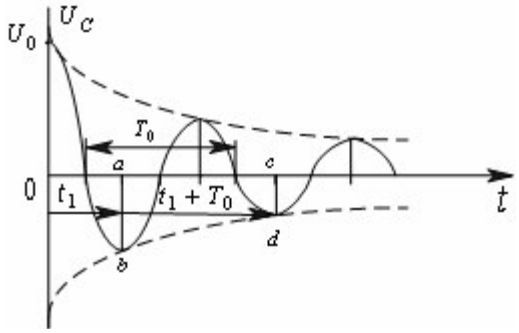



Рисунок 6. Колебательный переходной процесс

Тогда

$$\frac{u(t_1)}{u(t_1+T_c)} = \frac{U_{cm}e^{-\delta t_1}\sin(\omega_c t_1 + v)}{U_{cm}e^{-\delta(t_1+T_c)}\sin\left[\omega_c(t_1+T_c)+v\right]} = e^{\delta T_c} = \frac{ab}{cd}.$$

***Электрические дифференцирующие и интегрирующие цепи***

Дифференцирующей называется цепь, в которой выходная величина пропорциональна производной по времени от входной величины. Простейшей дифференцирующей цепью с использованием элементов R и C является схема рисунка 7, в которой при $R << \frac{1}{\omega C}$, $u_{вых} = u_R$.

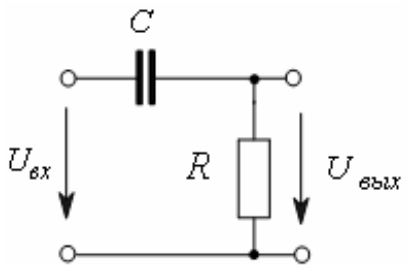


Рисунок 7. Дифференцирующая RC-цепь

Электрическое интегрирование можно осуществить при помощи схемы рисунка 8 при условии

$$R >> \frac{1}{\omega C},\ u_{вых} = u_c .$$

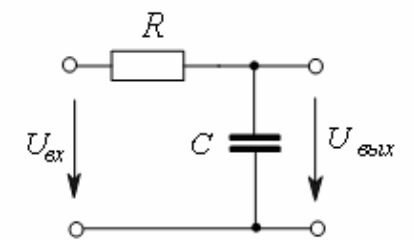



Рисунок 8. Интегрирующая RC-цепь

Дифференцирующая цепь с использованием элементов R и L показана на рисунке 9 при $R >> \omega L$, $u_{вых} = u_L$.

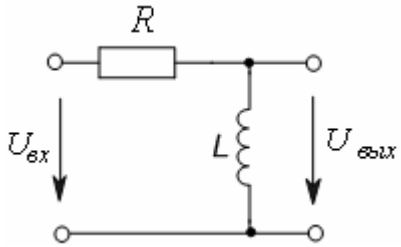



Рисунок 9. Дифференцирующая RL-цепь

Электрическое интегрирование можно осуществить также при помощи схемы рисунка 10 при условии $R << \omega L$, $u_{вых} = u_R$.

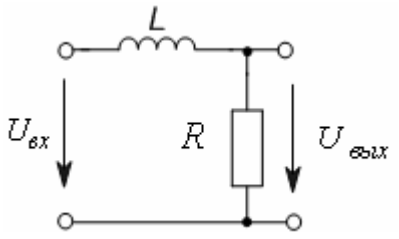



Рисунок 10. Интегрирующая RL-цепь

***Переходные и импульсные характеристики***

Переходная характеристика определяется как отношение реакции цепи на ступенчатое воздействие к величине этого воздействия при нулевых начальных условиях. Переходная характеристика численно совпадает с реакцией цепи на воздействие в виде единичной функции 1(t). Переходную характеристику h(t) можно определить, рассчитав переходный процесс и найдя $u_{вых}(t)$ при подключении к цепи источника постоянной э.д.с. E = 1 В.

Импульсная характеристика определяется как отношение реакции цепи на бесконечно короткий импульс бесконечно большой высоты и конечной площади к площади этого импульса при нулевых начальных условиях. Импульсная характеристика численно совпадает с реакцией цепи на воздействие в виде дельта-функции

$$\delta(t) = \frac{d1(t)}{dt} .$$

Взаимосвязь между переходной h(t) и импульсной $h_\delta(t)$ характеристиками определяется известными операторными выражениями:

$$h_\delta(t) = h(0)\cdot\delta(t) + h'(t) \doteqdot K(p) ;$$

$$h(t) = \int_0^t h_\delta(t)dt \doteqdot H(p) = \frac{K(p)}{p} ;$$

$K(p) = \frac{U_{вых}(p)}{U_{вх}(p)}$ – операторная передаточная функция цепи;

H(p) – изображение переходной характеристики.

Подставив в K(p) вместо p комплексную частоту jω, получим комплексную частотную характеристику цепи K(jω). Частотные зависимости модуля K(ω) и аргумента φ(ω) называют амплитудночастотной (АЧХ) и фазочастотной (ФЧХ) характеристиками цепи.

***Операторный метод расчета***

Для расчета переходного процесса операторным методом составляем операторную схему замещения цепи после коммутации (рисунок 11), в которой:

1) источник постоянного напряжения E заменен изображением $E(p) = \frac{E}{p}$;

2) индуктивность с начальным током i(0) заменяем операторным сопротивлением $Z_1(p) = pL$ и внутренним источником э.д.с. L*i(0), направленным согласно с током в индуктивности;

3) емкость заменяем операторным сопротивлением $\frac{1}{pC}$ и внутренним источником э.д.с. $\frac{u_C(0)}{p}$, направленным встречно току в емкости;

4) составляем операторные уравнения и определяем изображения искомых токов и напряжений;

5) используя обратное преобразование Лапласа или теорему разложения, находим оригиналы искомых токов и напряжений.

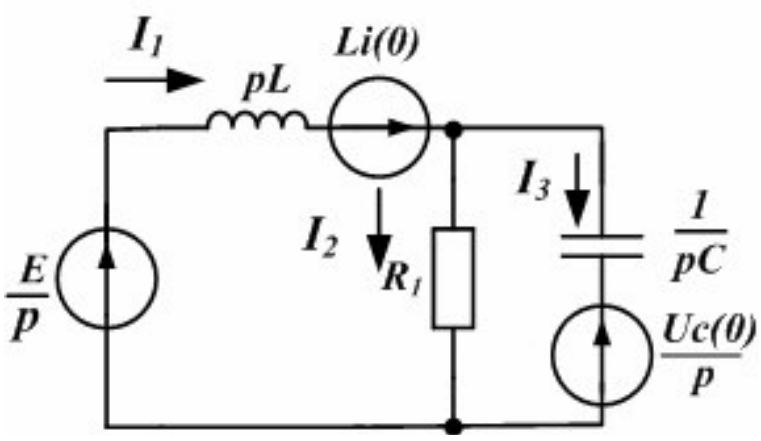


Рисунок 11. Операторная схема замещения цепи

Операторный метод дает более компактное решение и удобен для расчета цепей высоких порядков, содержащих более двух реактивных накопительных элементов.

***Лабораторное задание***

Наблюдение переходных процессов затруднено их кратковременностью. Поэтому для исследования применяется входной сигнал в виде периодической последовательности прямоугольных импульсов.

На рисунке 12 изображена схема измерений в реальной лаборатории МЭЛ.

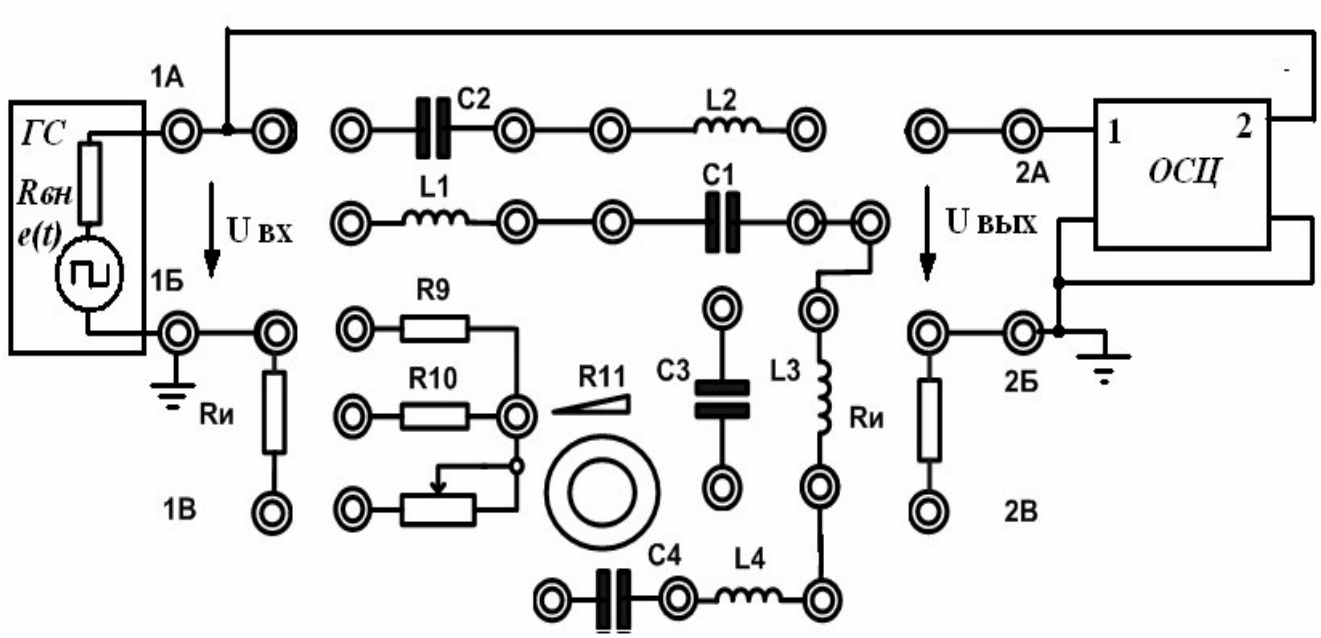



Рисунок 12. Схема измерений на МЭЛ

Генератор сигналов надо установить в режим формирования прямоугольных импульсов. Амплитуду импульсов следует установить равной 1 В. В качестве генератора прямоугольных импульсов в лабораторной установке может использоваться внутренний функциональный генератор или внешний генератор. Эквивалентная схема генератора состоит из импульсного источника напряжения с внутренним сопротивлением Rвн. Форму входных и выходных импульсов наблюдают на двухканальном осциллографе. Исследуемые цепи собирают из пассивных элементов второй панели МЭЛ.

А. Настройка функционального генератора.

1. Собрать схему измерений (см. рисунок 12).
2. Установить частоту генератора 5 кГц. Включить осциллограф. Установить режим внутренней синхронизации по 2-му каналу. Установить скважность импульсов $N = \frac{T}{t_{имп}} = 2$ (Т – период повторения импульсов, $t_{имп}$ – длительность импульса высокого уровня). Двойную амплитуду выходных импульсов в режиме холостого хода генератора $u_{xx}$ установить равной 1 В.

$$R_{вн} = \frac{(u_{xx} - u_{R10}) \cdot R_{10}}{u_{R10}}. \qquad (15)$$

B. Исследование RC-цепи.

1. Собрать интегрирующую RC-цепь по схеме рисунка 8, включив заданную преподавателем емкость и сопротивление $R_{10}$ = 100 Ом. Вход RC-цепи подключить к клеммам 1А и 1Б генератора сигналов. Выход RC-цепи подключить к клеммам 2А и 2Б схемы измерений.

2. Длительность развертки осциллографа установить такой, чтобы на экране наблюдались не более двух периодов импульсного сигнала.

3. Снимая выходной сигнал с емкости, наблюдать и зарисовать осциллограммы напряжений $u_{вх}$ и $u_{вых} = u_C(t)$, соблюдая масштабы напряжения и времени. Измерить по осциллограмме напряжения $u_C(t)$ постоянные времени RC-цепи при заряде емкости $\tau_з$ и разряде емкости $\tau_р$.

4. Включить вместо $R_{10}$ сопротивление $R_9$ = 1 кОм. Повторить исследования по п. 3.

5. Собрать дифференцирующую RC-цепь рисунка 7, включив ту же емкость, что в п. 1, и сопротивление $R_{10}$ = 100 Ом. Повторить исследования по п. 3.

C. Исследование RL-цепи.

1. Собрать интегрирующую RL-цепь по схеме рисунка 9, используя заданную преподавателем индуктивность и сопротивление $R_{10}$ = 100 Ом. Провести исследования по п. 3 (раздел В), снимая выходной сигнал с сопротивления.

2. Включить вместо $R_{10}$ сопротивление $R_9$ = 1 кОм. Повторить исследования по п. 3 (раздел В).

3. Собрать дифференцирующую RL-цепь по схеме рисунка 9, используя заданную преподавателем индуктивность и сопротивление $R_{10}$ = 100 Ом. Провести исследования по п. 3 (раздел В), снимая выходной сигнал с индуктивности.

4. Включить вместо $R_{10}$ сопротивление $R_9$ = 1 кОм. Повторить исследования по п. 3 (раздел В).

D. Исследование RLC-цепи.

1. Установить частоту генератора 500 Гц. Собрать RLC-цепь, используя емкость и индуктивность из предыдущих пунктов исследования. Активное сопротивление составить из последовательного соединения резистора $R_{10}$ = 100 Ом и переменного резистора $R_{11}$. Установить наибольшее значение резистора $R_{11}$, равное 2 кОм. Выходной сигнал снимать с резистора $R_{10}$. Зарисовать осциллограммы входного напряжения и выходного напряжения $u_{R10}(t)$, пропорционального току в цепи.

2. Изменяя сопротивление резистора $R_{11}$, наблюдать изменение формы тока в цепи. Зарисовать осциллограммы для критического случая переходного процесса, когда

$$R_{10} + R_{11} + R_{вн} = 2\sqrt{\frac{L}{C}} \ .$$

3. Измерив осциллографом отношение напряжений на резисторе $R_{10}$ и последовательном соединении ($R_{10}$ + $R_{11}$), рассчитать значение сопротивления потерь (с учетом внутреннего сопротивления генератора), при котором наблюдается критический переходный процесс.

4. Уменьшить до нуля сопротивление $R_{11}$. Наблюдать колебательный переходный процесс. Зарисовать осциллограммы напряжения $u_{R10}(t)$.

E. Исследование переходных и импульсных характеристик.

1. Установить частоту повторения 100 Гц и двойную амплитуду импульсов 1 В.

2. По указанию преподавателя собрать одну из схем рисунков 7–10. Зарисовать осциллограммы выходного сигнала, соответствующие переходной характеристике цепи.

3. Установить максимальную скважность импульсного сигнала, уменьшив до минимума длительность импульса высокого уровня. Зарисовать осциллограммы выходного сигнала $u_{вых}(t)$. Приближенно импульсная характеристика цепи при действии короткого импульса единичной амплитуды для $t > t_{имп}$ определяется соотношением

$$h_{\delta}(t) \approx \frac{1}{t_{имп}} u_{вых}(t) .$$

**Лабораторная работа № 7. Электрические цепи с магнитно-связанными катушками**

***Продолжительность: 180 минут.***
***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 4.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

Цель работы – определение параметров магнитно-связанных катушек, изучение распределения токов, напряжений и мощностей в цепях с взаимной индуктивностью.

Рабоа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.
2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.
3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.
4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.
5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.
6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

*Описание набора элементов и приборов*

Пассивные элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

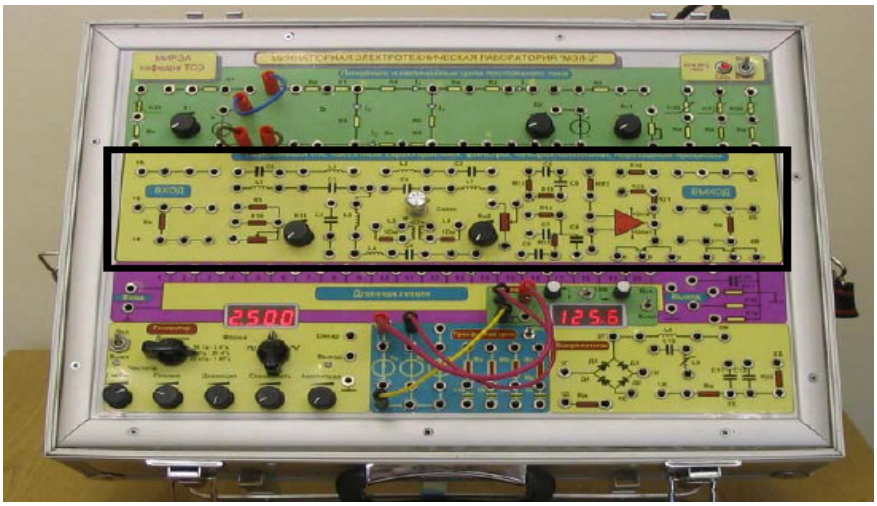


Рисунок 1. Панель МЭЛ для моделирования электрических цепей
с магнитно-связанными катушками

*Теоретические сведения и методы расчета электрических цепей с магнитно-связанными катушками*

Катушки называют магнитно-связанными, если они имеют общее магнитное поле и взаимно влияют друг на друга. При изменении тока в одной катушке за счет изменения общего магнитного поля во второй катушке наводится напряжение взаимной индукции.

На рисунке 2 катушки индуктивности $L_1$ и $L_2$ магнитно-связанные.

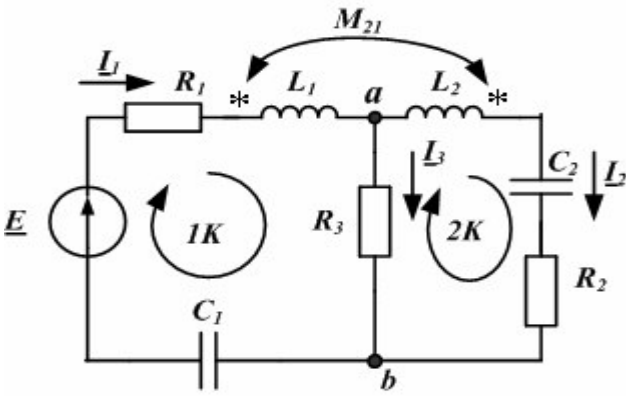



Рисунок 2. Схема цепи с взаимной индуктивностью

На схеме это обозначается стрелкой с указанием взаимной индуктивности катушек $M_{21}$. Взаимная индуктивность $M_{21}$ является коэффициентом пропорциональности между напряжением взаимной индукции, наводимым во второй катушке, и производной тока в первой катушке:

$$u_{2M}(t) = M_{21}\frac{di_1}{dt} \ .$$

Знак наводимого напряжения зависит от направления намотки катушек и направления токов в них. Если магнитные поля, создаваемые токами в катушках, складываются, такое включение катушек называют согласным и напряжения самоиндукции и взаимной индукции складываются. Если магнитные поля катушек вычитаются, включение катушек называют встречным и напряжения самоиндукции и взаимной индукции вычитаются. В электрических схемах у магнитно-связанных катушек обозначают «Одноименные зажимы», маркируя их звездочками или кружочками. Если токи в катушках одинаково направлены относительно одноименных зажимов, включение является согласным. Если токи в катушках направлены неодинаково относительно одноименных зажимов, включение является встречным. В схеме рисунка 2 катушки включены встречно. Для напряжения на второй катушке можно написать

$$u_2(t) = L_2 \frac{di_2}{dt} - M_{21} \frac{di_1}{dt} .$$

Это уравнение в символической форме для комплексных действующих значений записывается следующим образом:

$$\underline{U}_2 = j\omega L_2 \underline{I}_2 - j\omega M_{21} \underline{I}_1 .$$

В линейных электрических цепях по принципу взаимности $M_{21} = M_{12} = M$.

*Развязка магнитно-связанных цепей*

Развязкой называется замена магнитно-связанных цепей эквивалентными цепями без магнитных связей.

***Правила развязки***

1. Если одноименные зажимы двух магнитно-связанных индуктивностей одинаково расположены относительно узла (рисунок 3), то эти две индуктивности можно заменить эквивалентной схемой (рисунок 4) с тремя индуктивностями без магнитной связи.

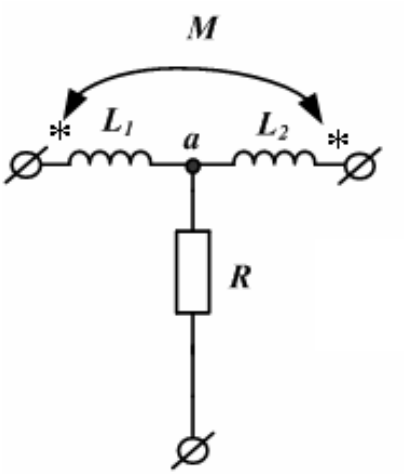



Рисунок 3. Две магнитно-связанные индуктивности,
одинаково расположенные относительно узла

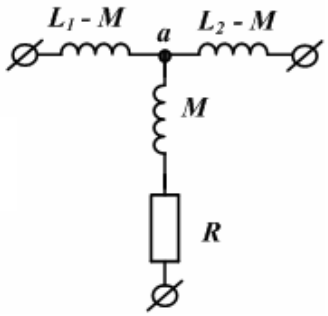



Рисунок 4. Эквивалентная схема с тремя индуктивностями без магнитной связи

Если одноименные зажимы двух магнитно-связанных индуктивностей неодинаково расположены относительно узла (рисунок 5), то эти две индуктивности можно заменить эквивалентной схемой (рисунок 6) с тремя индуктивностями без магнитной связи. Индуктивность с отрицательным значением (– M) имеет расчетный характер.

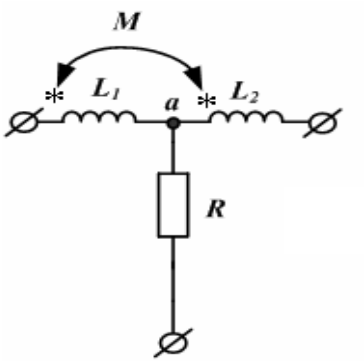



Рисунок 5. Зажимы двух магнитно-связанных индуктивностей, неодинаково расположены относительно узла

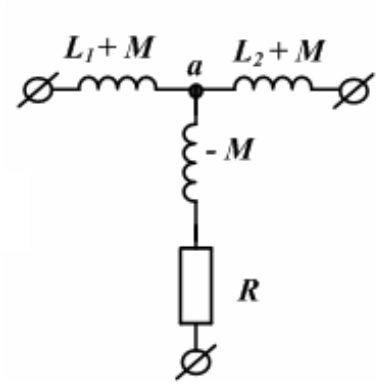



Рисунок 6. Эквивалентная схема с тремя индуктивностями без магнитной связи

***Расчет цепи методом развязки***

В схеме рисунка 2 одноименные зажимы катушек одинаково расположены относительно узла *a*. Поэтому для развязки применяем эквивалентную схему рисунка 4. Преобразованная схема без магнитных связей показана на рисунке 7.

На схеме рисунка 7 у каждого элемента указаны значения комплексных сопротивлений в Омах. Действующее значение источника напряжения равно 16 В.

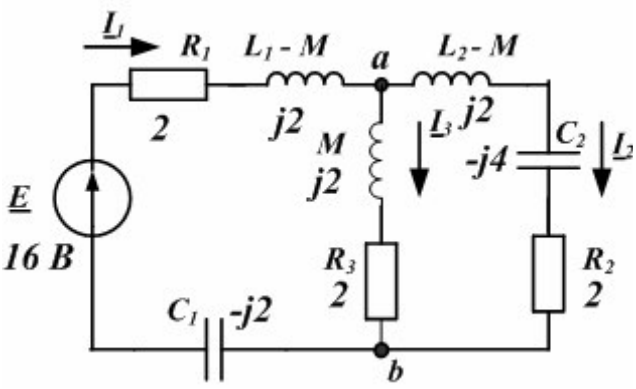



Рисунок 7. Эквивалентная схема цепи после развязки

Расчет схемы можно провести вручную. Находим эквивалентное сопротивление двух параллельных ветвей между узлами *a*-*b*:

$$\underline{Z}_{ab} = \frac{(2 + j2 - j4) \cdot (2 + j2)}{2 + j2 - j4 + 2 + j2} = 2\,\text{Ом}.$$

Находим входное сопротивление цепи:

$$\underline{Z}_{вх} = 2 + j2 + 2 - j2 = 4\,\text{Ом}.$$

Находим первый ток:

$$\underline{I}_1 = \frac{\underline{E}}{\underline{Z}_{вх}} = \frac{16}{4} = 4\,\text{А}.$$

Находим напряжение между узлами:

$$\underline{U}_{ab} = \underline{I}_1 \cdot \underline{Z}_{ab} = 8\,\text{В}.$$

Находим токи во второй и третьей ветви:

$$\underline{I}_2 = \frac{8}{2 - j2} = 2 + j2\,\text{А};$$

$$\underline{I}_3 = \frac{8}{2 + j2} = 2 - j2\,\text{А}.$$

***Определение параметров магнитно-связанных катушек***

В электротехнической лаборатории МЭЛ магнитно-связанные катушки $L_5$ и $L_6$ с измерительными сопротивлениями $R_и$ = 1 Ом (рисунок 8) находятся на второй панели наборного поля. Магнитная связь регулируется поворотом ручки "М" с градуировкой положения от 1 до 5. Для определения собственных параметров катушки $L_5$ ($Z_5$, $R_5$, $X_5$) на реальном макете надо собрать схему экспериментальных исследований рисунка 8, позволяющую измерить напряжение $U_1$, ток $I_1 = U_{и1}/1$ Ом, угол фазового сдвига φ между $U_1$ и $I_1$.

Катушка $L_6$ при этих измерениях должна быть разомкнута. По этим данным можно подсчитать:

$$Z_{BX} = \frac{U_1}{I_1};$$

$$R_5 = Z_{BX} \cdot \cos\varphi_{BX} - R_И;$$

$$X_5 = Z_{BX} \cdot \sin\varphi_{BX};$$

$$Z_5 = \sqrt{R_5{}^2 + X_5{}^2};$$

$$\varphi_5 = \text{arctg}\frac{X_5}{R_5};$$

$$\underline{Z}_5 = Z_5 \cdot e^{j\varphi_5}.$$

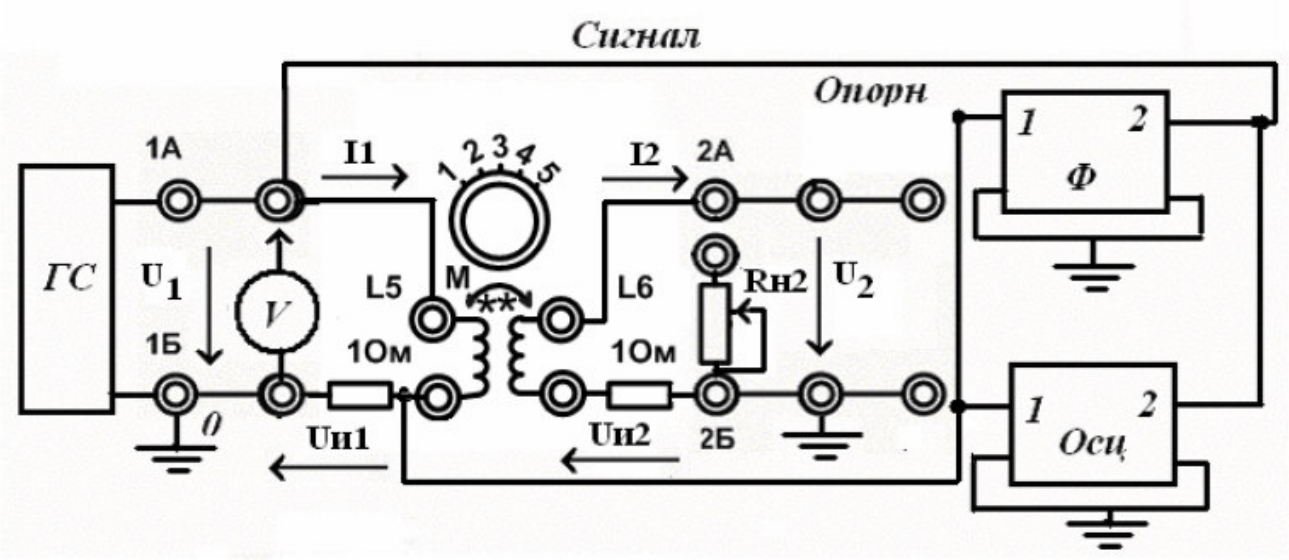



Рисунок 8. Схема экспериментальных исследований в МЭЛ

Для определения параметров $Z_6$, $R_6$, $X_6$ катушки $L_6$ надо в схеме рисунка 8 катушки $L_5$ и $L_6$ поменять местами и еще раз записать показания приборов $U_1$, $I_1$, $\varphi_{BX}$. Расчет параметров катушки $L_6$ проводится так же, как для $L_5$.

***Последовательное соединение магнитно-связанных катушек***

***Определение сопротивления магнитной связи***

Для определения сопротивления магнитной связи $X_M$ следует катушки $L_5$ и $L_6$ соединить последовательно и подключить к клеммам 1А и 1Б схемы рисунка 8. "Одноименные зажимы" катушек обозначены звездочками.

Схема на рисунке 9 соответствует согласному включению катушек, а схема на рисунке 10 соответствует встречному включению.

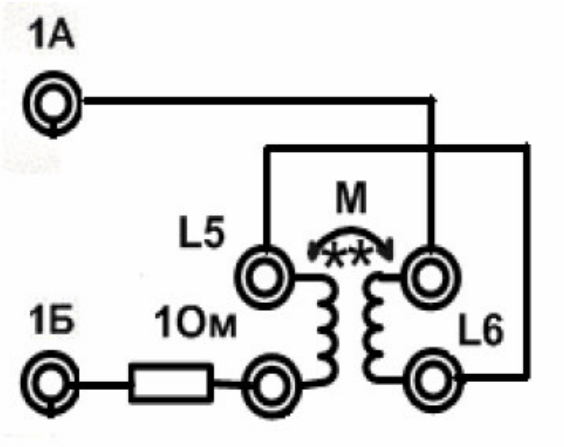



Рисунок 9. Согласное включение катушек

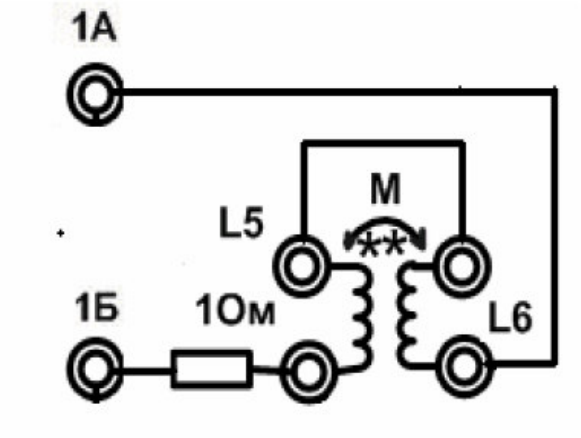



Рисунок 10. Встречное включение катушек

Для определения сопротивления магнитной связи следует при произвольно выбранном последовательном соединении катушек записать показания приборов. Затем поменять местами концы второй катушки и вновь записать показания приборов.

Для обоих опытов подсчитать значения:

$$Z_{BX} = \frac{U_1}{I_1};$$

$$X_1 = Z_{BX} \cdot \sin \varphi_{BX}.$$

Реактивное сопротивление при согласном включении $X_{согл}$ больше, чем при встречном включении $X_{встр}$. Сопротивление магнитной связи $X_M$ определяется по формуле

$$X_M = \omega M = \frac{X_{согл} - X_{встр}}{4}.$$

Здесь ω – круговая частота напряжения источника.

Сопротивление магнитной связи можно также определить в схеме (см. рисунок 8) другим способом. Для этого надо измерить ток $I_1$ и напряжение $U_2$ на разомкнутой второй катушке ($L_6$) и подсчитать

$$X_{M12} = \frac{U_2}{I_1}.$$

Если поменять катушки $L_5$ и $L_6$ местами и измерить значения тока $I_2$ и напряжения $U_1$ на разомкнутой обмотке первой катушки ($L_5$), то можно подсчитать

$$X_{M21} = \frac{U_1}{I_2}.$$

Эти эксперименты должны подтвердить, что $X_{M12} = X_{M21} = X_M$.

***Лабораторное задание***

А. Определение параметров катушек и сопротивления взаимной индукции.

1. Собрать схему рисунка 8.

2. Установить напряжение генератора на первичной обмотке порядка 1В. Частоту генератора установить в соответствии с формулой

$$f(\text{кГц}) = 2{,}5 + 0{,}5 \cdot N,$$

где N – число, задаваемое преподавателем. Ручку регулятора магнитной связи поставить в крайнее правое положение и не менять в последующих опытах. Значение измерительных сопротивлений $R_{и1} = R_{и2} = 1\text{Ом}$. Нагрузка $R_{и2}$ отключена. Записать показания приборов в таблицу 1 (замер 1).

Таблица 1

| | Эксперимент | | | | Расчет | | | |
|---|---|---|---|---|---|---|---|---|
| Замер 1 | $U_1$= | $U_и$= $I_1$= | φ = | $U_2$ = | $Z_5$= | $R_5$= | $X_5$ = | $X_{M12}$= |
| | | | | | | | | |
| Замер 2 | $U_2$= | $U_и$= $I_2$= | φ = | $U_1$ = | $Z_6$= | $R_6$= | $X_6$ = | $X_{M21}$ = |
| | | | | | | | | |
| Замер 3 | $U_1$= | $U_и$= $I_1$= | φ = | | $Z_э$= | $R_э$= | $X_э$ = | $X_{M12}$ = |
| | | | | | | | | |
| Замер 4 | $U_1$= | $U_и$= $I_1$= | φ = | | $Z_э$= | $R_э$= | $X_э$ = | $X_{M12}$ = |
| | | | | | | | | |
| Замер 5 | $U_1$= | $U_и$= $I_1$= | φ = | $U_2$ = $I_2$=0 | | | | |
| | | | | | | | | |
| Замер 6 | $U_1$= | $U_и$= $I_1$= | φ = | $U_2$ = $I_2$= | | | | |

3. Отключить питание генератора, поменять катушки местами и повторить эксперимент. Данные записать в таблицу 1 (замер 2).

4. Произвести расчет параметров обеих катушек и сопротивления взаимоиндукции. Результаты расчета записать в таблицу 1. Убедиться, что $X_{M12} = X_{M21} = X_M$.

В. Последовательное соединение катушек.

1. Начертить и собрать схему с двумя последовательно соединенными магнитносвязанными катушками. Приборы подключить согласно схеме рисунка 8.

2. При согласном последовательном соединении катушек (см. рисунок 9) записать показания приборов (замер 3).

3. Поменять местами концы второй катушки и при встречном последовательном соединении катушек (см. рисунок 10) вновь записать показания (замер 4).

Используя данные замеров 3 и 4, подсчитать полное сопротивление катушек в обоих случаях и, основываясь на этих данных, подтвердить, какой из опытов соответствует согласному и какой встречному включению. Убедиться в правильности разметки зажимов катушек на панели МЭЛ. Составить эскиз с обозначением зажимов катушек.

С. Исследование работы трансформатора.

1. Собрать схему (см. рисунок 8). Подключить к клемме 2А переменный резистор $R_{н2}$. Поддерживая на зажимах первичной обмотки трансформатора напряжение порядка 1В, записать показания приборов для двух режимов: а) $R_{н2} = \infty$(х.х.); б) $R_{н2} = 0{,}5\ R_{нмах}$. Результаты (замер 5 и замер 6) записать в таблицу 1.

2. Используя найденные экспериментально параметры катушек линейного трансформатора и сопротивление магнитной связи, составить схему замещения линейного трансформатора и выполнить теоретический расчет токов для $R_{н2} = 0{,}5\ R_{нмах}$.

**Лабораторная работа № 8. Исследование характеристик биполярного транзистора и усилителя на биполярном транзисторе**

***Продолжительность: 180 минут.***

***Дисциплина*: *«Электротехника, электроника и схемотехника». Юнита 5.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вступительная часть**

**Цель работы.** Исследование вольтамперных характеристик биполярного транзистора и усилителя на его основе.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

✓ заглавие (номер и название лабораторной работы);

✓ схемы исследованных электрических цепей;

✓ результаты исследований (в виде таблиц и графиков);

✓ расчетную часть задания;

✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Описание набора элементов и приборов***

Элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

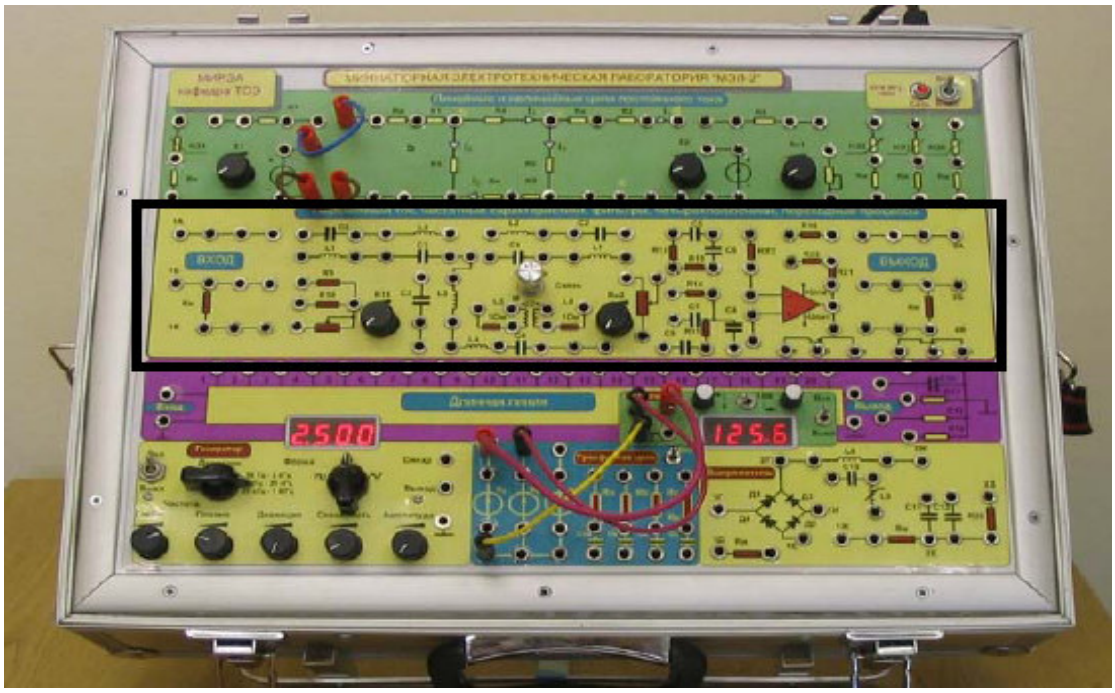



Рисунок 1. Панель МЭЛ для исследования характеристик биполярного транзистора и усилителя на биполярном транзисторе

**2 Теоретические сведения и методы расчета электрических цепей с биполярными транзисторами и усилителями на биполярном транзисторе**

Биполярным транзистором называют полупроводниковый прибор, имеющий два взаимодействующих между собой p-n-перехода. В зависимости от последовательности чередования областей с различным типом проводимости различают n-p-n-транзисторы и p-n-p-транзисторы. Транзистор называется биполярным потому, что физические процессы в нем связаны с движением носителей обоих знаков (свободных дырок и электронов). Трехслойная структура n-p-n-транзистора показана на рисунке 2.

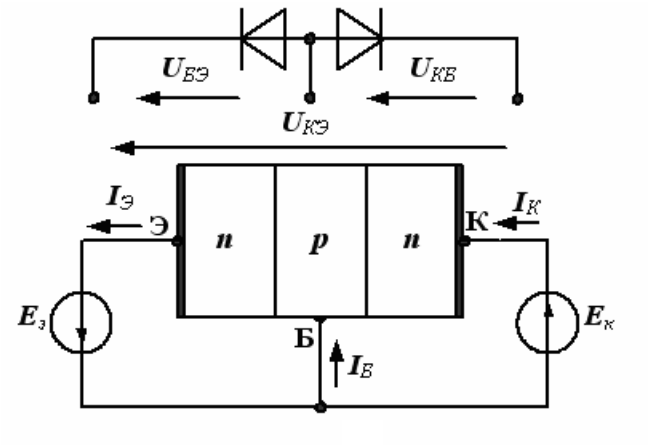



Рисунок 2. Структура n-p-n-транзистора

На рисунке 3 показано условное обозначение n-p-n-транзистора, на рисунке 4 – условное обозначение p-n-p-транзистора.

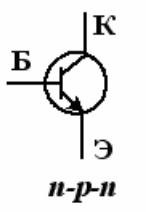



Рисунок 3. Условное обозначение n-p-n-транзистора

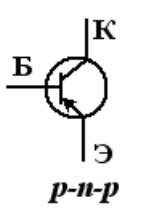



Рисунок 4. Условное обозначение p-n-p – транзистора

Средний слой биполярного транзистора называют базой «Б», один крайний слой называют коллектором «К», другой крайний слой называют эмиттером «Э». В зависимости от полярности напряжений, приложенных к электродам транзистора, различают следующие режимы его работы: линейный (усилительный), насыщения, отсечки, инверсный. В линейном режиме эмиттерный переход смещен в прямом направлении, а коллекторный - в обратном. В режиме насыщения оба перехода смещены в прямом направлении. В режиме отсечки оба перехода смещены в обратном направлении. В инверсном режиме коллекторный переход смещен в прямом направлении, а эмиттерный в обратном.

Биполярные транзисторы применяются в схемах усилителей, генераторов и преобразователей электрических сигналов, изготавливаются из кремния, германия или арсенида галлия и делятся на низкочастотные (до 3 МГц), среднечастотные (до 30 МГц), высокочастотные (до 300 МГц) и сверхвысокочастотные (более 300 МГц). По мощности транзисторы бывают маломощные(до 300 мВт), средней мощности (до 1,5 Вт) и большой мощности (более 1,5 Вт).

Работа транзистора основана на управлении токами электродов в зависимости от приложенных к его переходам напряжений. В линейном режиме приложенное к базе напряжение $U_{БЭ}$ (для n-p-n-транзистора $U_{БЭ} > 0$) открывает переход база-эмиттер. Свободные электроны инжектируются из эмиттера в базу, образуя ток эмиттера $I_Э$ в цепи эмиттера. Большая часть электронов, инжектированных из эмиттера в базу, втягивается

сильным электрическим полемр-n-перехода между базой и коллектором, образуя ток коллектора $I_К$ в цепи коллектора. Незначительная часть свободных электронов, инжектированных из эмиттера в базу, образуетток $I_Б$.

В схеме рисунка 2 база является общим электродом входной и выходной цепи. Такая схема включения биполярного транзистора называется схемой с общей базой (ОБ). Для усиления сигналов применяют также схемы включения биполярных транзисторов с общим коллектором (ОК) и общим эмиттером (ОЭ).

Схема с общим эмиттером наиболее распространена, исследуется в лабораторной работе и показана на рисунке 5.

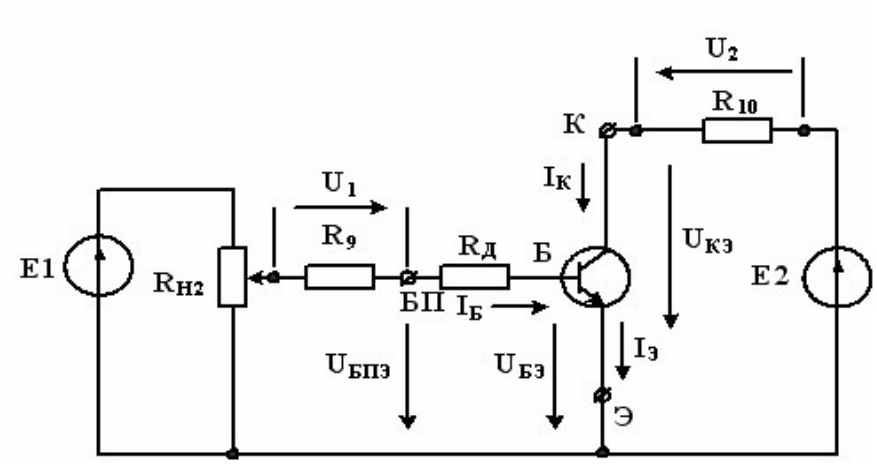



Рисунок 5. Схема включения транзистора с общим эмиттером

В этой схеме дополнительное сопротивление $R_Д$ = 100 Ом служит для защиты транзистора от пробоя перехода база-эмиттер, $R_9$ = 1 кОм, $R_{10}$ = 100 Ом, $U_{БПЭ}$ – напряжение между гнездом базы на панели стенда и гнездом эмиттера, $U_{БЭ}$ – напряжение база-эмиттер. Причем $U_{БЭ} = U_{БПЭ} -- I_Б*R_Д$.

В схеме ОЭ ток коллектора, ток базы и ток эмиттера связаны соотношениями:

$$I_Э = Iк + I_Б,\ I_К = \beta * I_Б,$$

где $\beta$ – статический коэффициент передачи тока в схеме с общим эмиттером. Работу транзистора, включенного по схеме с ОЭ, определяют по статическим входным (рисунок 6) и выходным (рисунок 7) вольтамперным характеристикам.

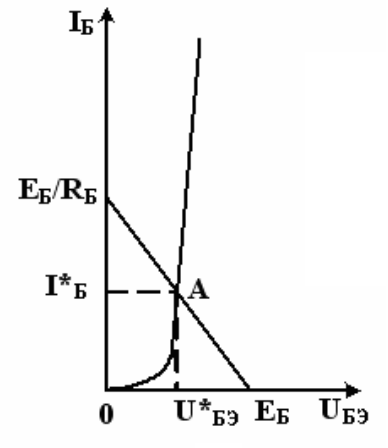



Рисунок 6. Входная характеристика биполярного транзистора

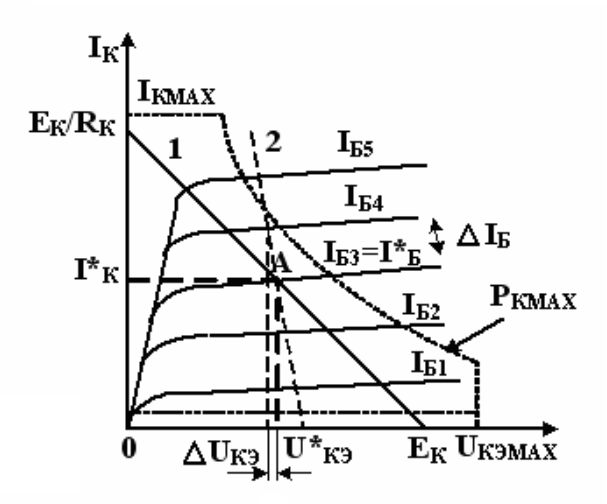



Рисунок 7. Выходные характеристики биполярного транзистора

Входная ВАХ является зависимостью тока базы от напряжения база-эмиттер при фиксированном напряжении коллектор-эмиттер. Выходные ВАХ являются зависимостями тока коллектора от напряжения коллектор-эмиттер при различных значениях тока базы. Вольтамперные характеристики биполярного транзистора показывают, что в линейной обрасти ток коллектора почти не зависит от напряжения коллектор-эмиттер и приращение тока пропорционально изменению тока базы. Ток базы и напряжение база-эмиттер почти не зависят от напряжения коллектор-эмиттер (на рисунке 6 все входные ВАХ заменены одной). При напряжении отсечки $U_{БЭ0}$ ток базы считают равным нулю.

В линейном режиме усиления малого сигнала биполярный транзистор описывают системой уравнений четырехполюсника в Н-параметрах:

$$\begin{aligned} u_{БЭ} &= h_{11}\cdot i_Б + h_{12}\cdot u_{КЭ};\\ i_К &= h_{21}\cdot i_Б + h_{22}\cdot u_{КЭ}, \end{aligned} \qquad (1)$$

где

$$\begin{aligned} h_{11} &= \frac{\Delta u_{БЭ}}{\Delta i_Б}\Big|_{u_{КЭ}=\text{const}}, \quad h_{12} = \frac{\Delta u_{БЭ}}{\Delta u_{КЭ}}\Big|_{i_Б=\text{const}},\\ h_{21} &= \frac{\Delta i_К}{\Delta i_Б}\Big|_{u_{КЭ}=\text{const}}, \quad h_{22} = \frac{\Delta i_К}{\Delta u_{КЭ}}\Big|_{i_Б=\text{const}}. \end{aligned} \qquad (2)$$

Н-параметры биполярного транзистора, которые можно рассчитать по вольт-амперным характеристикам и определить экспериментально. Их типовые значения находятся в пределах:

$$h_{11} = 10^3 - 10^4\,Ом, \quad h_{12} = 2\cdot 10^{-4} - 2\cdot 10^{-3},$$
$$h_{21} = 20 - 200, \quad h_{22} = 10^{-5} - 10^{-6}\,См.$$

Пренебрегая малым значением параметра $h_{12}$, получим схему замещения биполярного транзистора, включенного по схеме с ОЭ в режиме малого сигнала (рисунок 8).

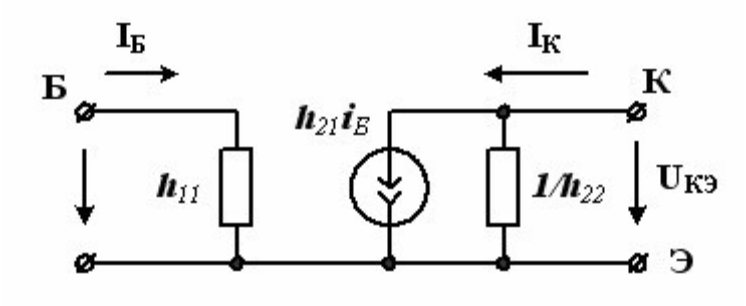



Рисунок 8. Схема замещения биполярного транзистора на постоянном токе и низких частотах

В этой схеме $h_{11} = R_{ВХ}$, $1/h_{22} = R_{ВЫХ}$ – входное и выходное сопротивления; $h_{21} * i_Б$ - источник тока, управляемый током базы $i_Б$.

Таким образом, биполярный транзистор представляет собой источник тока, управляемый током.

Эта схема замещения используется на постоянном токе и низких частотах, когда инерционность транзистора можно не учитывать. В более общем случае Н-параметры транзистора являются комплексными величинами, в схему замещения добавляются емкости между базой и коллектором $C_К$ и базой и эмиттером $C_Э$.

Для работы в линейном режиме на выходных характеристиках транзистора (см. рисунок 7) в режиме покоя выбирают рабочую точку А в центре линии нагрузки цепи коллектора 1. В рабочей точке по выходным характеристикам находят ток коллектора $I^*_К$ и ток базы $I^*_Б$. Область рабочих режимов транзистора на рисунке 7 отмечена пунктирными линиями и ограничивается максимальными допустимыми значениями тока коллектора $I_{КМАХ}$, напряжения $U_{КМАХ}$, мощности рассеяния $P_{КМАХ} \approx U_{КЭ}I_{КЭ}$ и нелинейными искажениями при малых значениях тока коллектора.

Для стабилизации рабочей точки в линейных усилительных каскадах обычно применяют схему с общим эмиттером и отрицательной обратной связью. Такая схема исследуется в лабораторной работе и показана на рисунке 9.

В МЭЛ-2 $R_8 = 510$ Ом, $R_{10} = 100$ Ом, $R_{14} = 10$ кОм, $R_{11} = 2{,}2$ кОм, $R_A = R_D = 2$ кОм, $C_A = C_B == C_C = 2{,}2$ мкФ.

Если напряжение входного сигнала $u_{вх}$ от генератора сигналов ГС невелико, то работу усилительного транзисторного каскада можно представить в виде наложения режима покоя с постоянным источником ЭДС $E_К$ и с постоянными составляющими тока базы $I^*_Б$, тока коллектора $I^*_К$ и тока эмиттера $I^*_Э$, соответствующими точке А на рабочей характеристике, и режима малого сигнала с переменными составляющими $i_Б$, $i_К$, $u_{ВХ}$, $u_{ВЫХ}$.

В режиме покоя рабочая точка находится на пересечении нагрузочной прямой:

$$I_К = \frac{E - U_{КЭ}}{R_К + R_Э}. \qquad (3)$$

Сопротивление $R_Э = R_{10}$ создает отрицательную обратную связь и стабилизирует режим покоя.

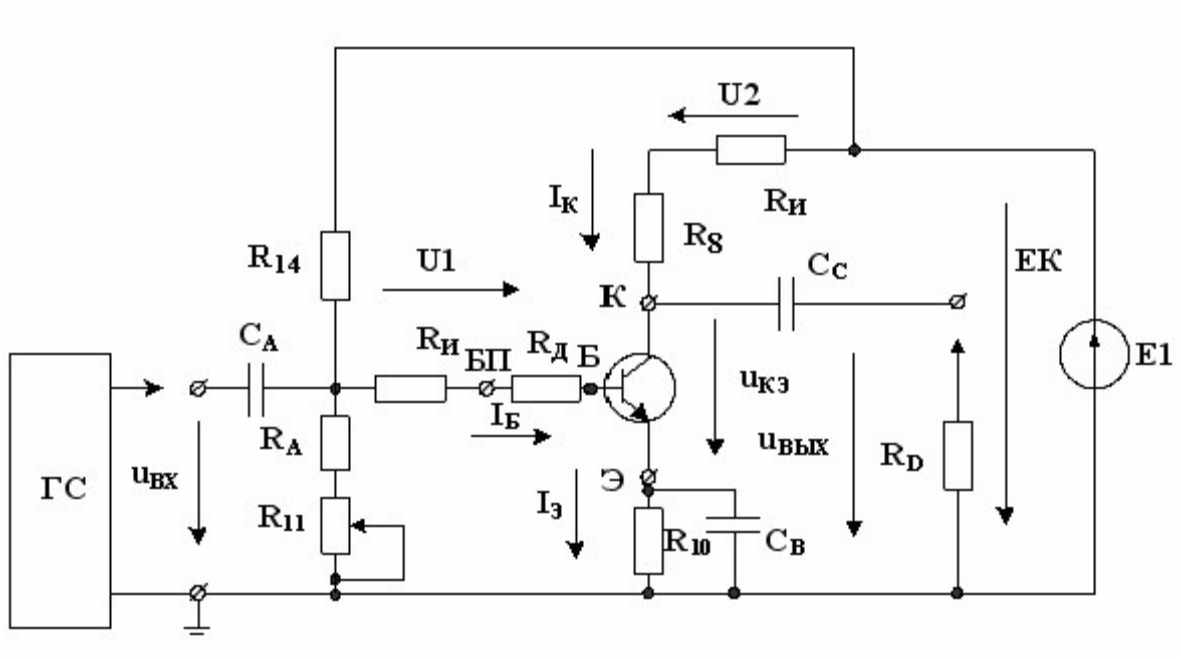



Рисунок 9. Схема усилительного транзисторного каскада с общим эмиттером

Схема замещения режима малого сигнала на низких частотах показана на рисунке 10.

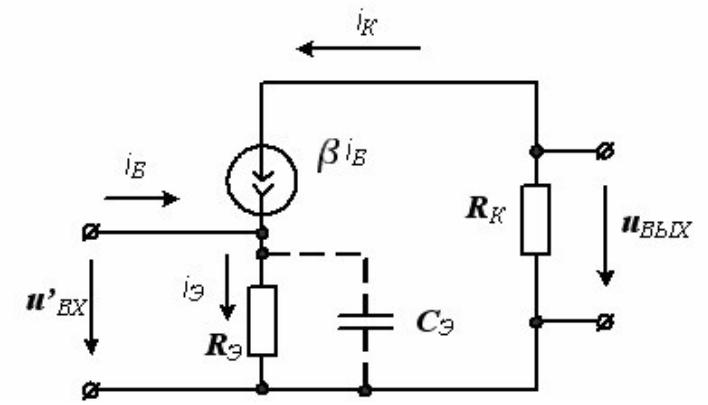



Рисунок 10. Схема замещения усилительного транзисторного каскада
для малого переменного сигнала

В схеме замещения (см. рисунок 10) сопротивления $R_{14}$ и $R_{11} + R_А$ схемы (см. рисунок 9) не учитываются, $R_К$ соответствует $R_8$. Емкости для переменного сигнала сначала считаются короткозамкнутыми, сопротивления $R_И$ не учитываются. Для схемы замещения без учета емкостей коэффициент усиления по напряжению в режиме холостого хода

$$\underline{K}'_{Ux} = \frac{U_{ВЫХ}}{U_{ВХ}} = -\frac{R_К}{R_Э + r_Э}, \qquad (4)$$

где $r_Э = \dfrac{25\,мВ}{I_Э}$ – дифференциальное сопротивление перехода база-эмиттер, $I_Э$ - постоянный ток эмиттера.

Отрицательное значение комплексного коэффициента усиления напряжения отражает изменение фаз выходного напряжения на $180^o$ относительно входного напряжения.

Если в схеме учесть емкость $C_Э$, то коэффициент усиления в режиме холостого хода станет равным

$$K_{Ux} = -\frac{R_K}{R_Э + r_э}\sqrt{1 + (\omega C_Э R_Э)^2}\ . \qquad (5)$$

Входное сопротивление по переменному току определяется как параллельное соединение входного сопротивления транзистора $r_{БЭ} = h_{11} = \beta r_Э$ и сопротивления $R_Б$, которое служит для установки рабочей точки каскада.

В схеме (см. рисунок 9)

$$R_Б = \frac{R_{14}(R_A + R_{11})}{R_{14} + R_A + R_{11}},\ R_{BX} = \frac{(r_{БЭ} + R_Д)R_Б}{r_{БЭ} + R_Д + R_Б}\ . \qquad (6)$$

Входная разделительная емкость $C_A$ образует с входным сопротивлением $R_{BX}$ делитель напряжения и коэффициент передачи входной цепи составит

$$K_{ВЦ} = \frac{R_{BX} \cdot \omega C_A}{\sqrt{1 + (R_{BX} \cdot \omega C_A)^2}}\ . \qquad (7)$$

С учетом (5), (7) коэффициент усиления транзисторного каскада с общим эмиттером на низких частотах можно рассчитать по формуле

$$K_{ВЦ} = \frac{R_{BX} \cdot \omega C_A}{\sqrt{1 + (R_{BX} \cdot \omega C_A)^2}}\ . \qquad (8)$$

С учетом сопротивления нагрузки $K_{ВЦ} = \frac{R_{BX} \cdot \omega C_A}{\sqrt{1 + (R_{BX} \cdot \omega C_A)^2}}$ для малого переменного сигнала на высокой частоте соответствует нагрузочная прямая 2, показанная на рисунках 6-7 пунктирной линией и определяемая уравнением

$$u_{КЭ} = -\frac{R_K R_H}{R_H + R_K} i_K\ . \qquad (9)$$

Ток в цепи нагрузки равен

$$i_H = -\frac{R_K}{R_H + R_K} i_K\ . \qquad (10)$$

На высоких частотах применяют более точные модели транзисторов. Наиболее распространенными являются модели, основанные на схеме замещения Джиаколетто (рисунок 11), в которой сопротивление $r_б$ – распределенное сопротивление базы, $g_э$ и $C_э$ – отражают полную проводимость эмиттерного перехода, $g_к$ и $C_к$ – учитывают влияние коллекторного перехода, проводимость $g_{кэ}$ учитывает связь между эмиттером и коллектором. Усилительные свойства транзистора учтены крутизной S.

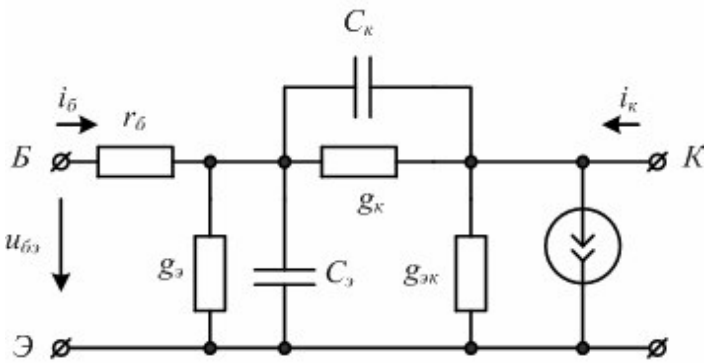



Рисунок 11. Схема замещения транзистора на высокой частоте

***Лабораторное задание***

A. Исследование входной характеристики биполярного транзистора в схеме с общим эмиттером и определение статического коэффициента передачи тока.

1. Собрать на стенде МЭЛ схему, показанную на рисунке 5. В этой схеме использован транзистор КТ603.
2. В схеме рисунка 5 установить $E_1 = 2$ В, $R_{H2}$ в крайнее левое положение, $E_2 = 10$ В.
3. Увеличивая в схеме рисунка 5 значение $R_{H2}$ и при необходимости $E_1$, измерять и устанавливать заданные в таблице 1 значения напряжения $U_1$ и соответствующие им напряжения $U_{БПЭ}$ и $U_2$.

Результаты измерений записать в таблицу 1.

Таблица 1

| $U_1$, В | $U_{БПЭ}$, В | $I_Б$, мА | $U_{БЭ}$, В | $U_2$, В | $I_К$, мА | $\beta$ |
|---|---|---|---|---|---|---|
| 0 | | | | | | |
| 0,25 | | | | | | |
| 0,5 | | | | | | |
| 0,75 | | | | | | |
| 1 | | | | | | |
| 1,5 | | | | | | |
| 2 | | | | | | |

4. По данным измерений рассчитать и внести в таблицу значения:

$$I_Б = \frac{U_1}{R_9};\ U_{БЭ} = U_{БПЭ} - I_Б \cdot R_Д;\ I_К = \frac{U_2}{R_И};\ \beta = \frac{I_К}{I_Б}.$$

5. Построить график входной характеристики биполярного транзистора и зависимость $\beta(I_Б)$.

B. Исследование выходных характеристик биполярного транзистора в схеме с общим эмиттером.

1. В схеме измерений по п. А-2 установить ток базы $I_Б = \frac{U_1}{R_9} = 0{,}25$ мА. Изменяя $E_2$, провести измерения зависимости тока коллектора $I_К$ от напряжения коллектор-эмиттер $U_{КЭ}$. Результаты записать в таблицу 2.

Таблица 2

| | | $U_{КЭ}$, В | 0 | 2 | 4 | 6 | 8 | 10 |
|---|---|---|---|---|---|---|---|---|
| $I_Б$, мА | 0,25 | $I_К$, мА | | | | | | |
| | 0,5 | $I_К$, мА | | | | | | |
| | 0,75 | $I_К$, мА | | | | | | |
| | 1 | $I_К$, мА | | | | | | |
| | 1,5 | $I_К$, мА | | | | | | |
| | 2 | $I_К$, мА | | | | | | |

2. Выполнить измерения для других значений тока базы, указанных в таблице 2.

3. По данным таблицы 2 построить семейство выходных характеристик биполярного транзистора.

C. Выбор рабочей точки транзисторного каскада с общим эмиттером.

1. Для схемы транзисторного усилителя (см. рисунок 9) построить на семействе выходных характеристик линию нагрузки по постоянному току по формуле (3). Напряжение питания $E = 10$ В. Выбрать на линии нагрузки рабочую точку А, в которой $U_{КЭ} = E_К/2$. Определить для точки А постоянный ток базы $I^*_Б$ и постоянный ток коллектора $I^*_К$.

2. Собрать на стенде МЭЛ схему транзисторного усилителя с общим эмиттером (см. рисунок 9). Переменный входной сигнал не подключать.

3. Регулируя $R_{11}$ и измеряя ток базы, установить $I_Б = I^*_Б$. Измерить и записать значения постоянной составляющей тока коллектора $I_К$ и напряжения $U_{КЭ}$. Сравнить полученные значения с рассчитанными в точке А.

4. Выполнить повторную регулировку $R_{11}$ и установить напряжение $U_{КЭ} = 5$ В.

D. Исследование работы транзисторного усилителя с общим эмиттером в режиме малого сигнала.

1. Установить в функциональном генераторе частоту синусоидального сигнала 1 кГц, амплитуду входного сигнала $u_{ВХ} = 200$ мВ установить по осциллографу. Подключить входной сигнал к транзисторному усилителю.

2. Осциллографом наблюдать сигнал на выходе усилителя в режиме холостого хода без подключенной нагрузки. Если выходной сигнал не имеет существенных отличий от синусоидальной формы, измерить осциллографом амплитуду выходного сигнала. Если форма выходного сигнала существенно искажена, уменьшить амплитуду входного сигнала до 100 мВ. Зарисовать осциллограммы входного и выходного сигналов. Записать измеренное значение $u_{ВЫХ}$. Рассчитать коэффициент усиления по напряжению:

$$K_{Ux} = \frac{U_{ВЫХ}}{U_{ВХ}} .$$

3. Снять амплитудно-частотную характеристику транзисторного усилителя в режиме усиления малого сигнала, изменяя частоту входного сигнала в диапазоне от 200 Гц до 20 кГц. Результаты записать в таблицу 3.

Таблица 3

$U_{ВХ}$ = мВ

| $f$, кГц | 0,2 | 1 | 2 | 5 | 10 | 20 |
|---|---|---|---|---|---|---|
| $U_{ВЫХ}$ | | | | | | |
| $K_U(f)$ | | | | | | |

4. Подключить к транзисторному усилителю нагрузку $R_D$. Повторить измерения по п. D-3.

E. Исследование искажений выходного сигнала.

1. Установить частоту входного сигнала 20 кГц, напряжение 200 мВ. Наблюдая форму выходного сигнала, увеличить амплитуду входного сигнала до появления заметных искажений выходного сигнала. Зарисовать осциллограммы входного и выходного сигнала и записать значение напряжения входного сигнала $u_{ВХ\ МАХ}$.

2. Установить частоту и амплитуду входного сигнала по п. D-1. Изменяя сопротивление $R_{11}$, наблюдать появление искажений формы выходного сигнала. Зарисовать осциллограмму выходного сигнала. Отключить входной сигнал и измерить ток базы, ток коллектора и напряжение $u_{КЭ}$.

**Лабораторная работа № 9. Исследование характеристик полевого транзистора и усилителя на полевом транзисторе**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 5.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

***Цель работы.*** Исследование вольтамперных характеристик полевого транзистора и усилителей на его основе.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

***Описание набора элементов и приборов***

Элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

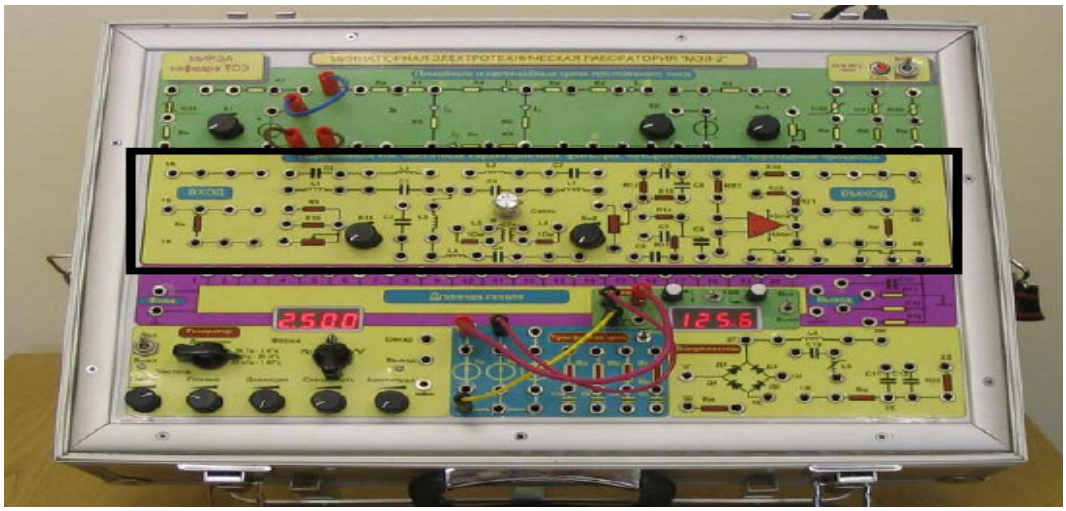


Рисунок 1. Панель МЭЛ для исследования характеристик полевого транзистора и усилителя на полевом транзисторе

**2 Теоретические сведения и методы расчета электрических цепей с полевым транзистором и усилителем на полевом транзисторе**

Полевыми или униполярными транзисторами называются полупроводниковые приборы, в которых изменение тока производится изменением проводимости проводящего канала с помощью электрического поля, перпендикулярного направлению тока. Прохождение тока в канале только одним типом зарядов. Электроды, подключенные к каналу, называются стоком (Drain) и истоком (Source). Управляющий электрод называется затвором (Gate). Напряжение управления прикладывается между затвором и истоком.

В зависимости от выполнения затвора униполярные транзисторы делятся на две группы: с управляющим p–n-переходом и с изолированным затвором на основе конструкции металл-диэлектрик-полупроводник (так называемые МДП-транзисторы).

Устройство полевого транзистора с управляющим p-n-переходом показано на рисунке 2.

Между истоком И и стоком С расположен n-канал из полупроводника n-типа и включен источник напряжения положительным полюсом к стоку. В n-канале есть ток проводимости $I_C$, значение которого зависит от сопротивления канала, связанного с его шириной. Ширину канала можно изменять, включив между затвором З и истоком И источник управляющего напряжения $E_З$ отрицательным полюсом к затвору. Передаточные характеристики полевых транзисторов, которые выражают зависимость тока стока от напряжения затвор-исток $I_C(U_{ЗИ})$, показаны на рисунке 3.

Устройство полевого транзистора с изолированным затвором показано на рисунке 4.

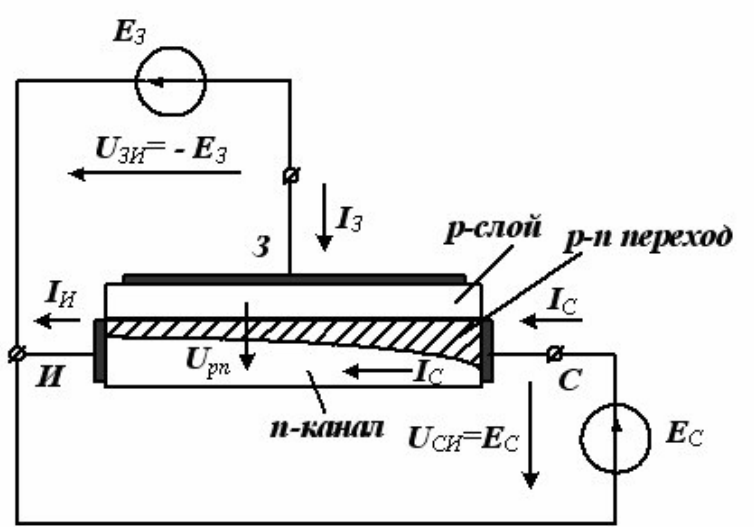



Рисунок 2. Устройство полевого транзистора с управляющим *p-n*-переходом

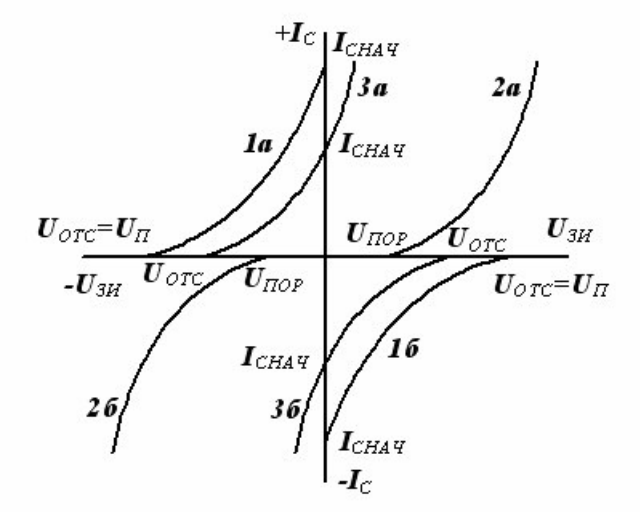



Рисунок 3. Передаточные характеристики полевых транзисторов разных типов

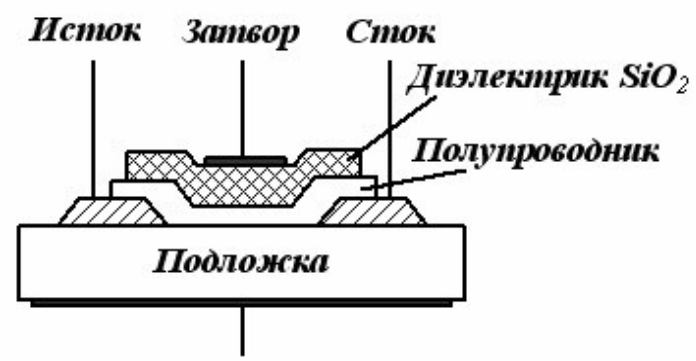



Рисунок 4. Устройство полевого транзистора с изолированным затвором

В полевых транзисторах с изолированным затвором электрод затвора изолирован от полупроводникового канала с помощью слоя диэлектрика из диоксида кремния $SiO_2$. Ток утечки затвора пренебрежимо мал. Полупроводниковый канал может быть обеднен носителями заряда или обогащен ими. При обедненном канале электрическое поле затвора повышает его проводимость, поэтому канал называется индуцированным. Если канал обогащен носителями, то он называется встроенным. Электрическое поле в этом случае в зависимости от полярности напряжения $U_{ЗИ}$ может приводить либо к обеднению канала носителями зарядов, либо к обогащению его. В результате изменяется проводимость канала.

Проводимость канала может быть электронной или дырочной. Если канал имеет электронную проводимость, то он называется n-каналом. Каналы с дырочной проводимостью называются p-каналами. Подложка П является полупроводником, отличающимся по проводимости от канала. Как правило, подложку соединяют с истоком.

Схематические изображения полевых транзисторов показаны на рисунке 5.

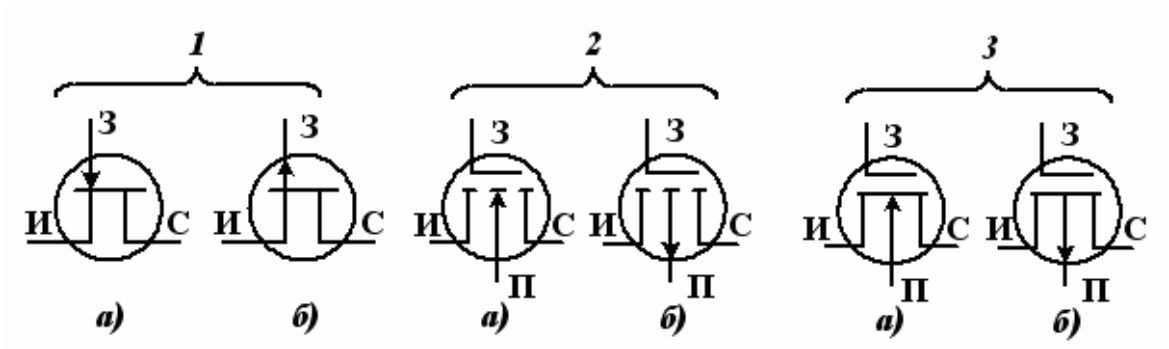



Рисунок 5. Схематические изображения полевых транзисторов:
*1* – с управляющим p-n-переходом; *2* – с индуцированным каналом;
*3* – со встроенным каналом; *а* – для канала n-типа, *б* – для канала p-типа

Важное значение имеют передаточные характеристики полевых транзисторов, позволяющие определить полярность управляющего напряжения, направление тока в канале и диапазон изменения управляющего напряжения (см. рисунок 3).

Полевые транзисторы с каналом n-типа имеют положительный ток и работают при положительном напряжении на стоке, а полевые транзисторы с каналом p-типа имеют отрицательный ток и работают при отрицательном напряжении на стоке. Характеристики полевых транзисторов с управляющим p-n-переходом при нулевом напряжении $U_{ЗИ}$ имеют максимальное значение тока $I_{СНАЧ}$. При увеличении запирающего напряжения ток стока уменьшается и при напряжении отсечки $U_{ОТС}$ становится близким к нулю.

Характеристики транзисторов с индуцированным каналом при нулевом напряжении на затворе имеют нулевой ток. Ток стока появляется при напряжении на затворе больше порогового и увеличивается с ростом напряжения $U_{ЗИ}$.

Характеристики транзисторов со встроенным каналом при нулевом напряжении на затворе имеют начальное значение тока $I_{СНАЧ}$. Эти транзисторы работают как при положительных, так и при отрицательных напряжениях на затворе.

Выходные характеристики МДП-транзистора с индуцированным каналом n-типа показаны на рисунке 6.

В линейной области полевой транзистор используется как сопротивление, управляемое напряжением на затворе, а в области насыщения – как усилительный элемент. Усилительные свойства определяются крутизной вольтамперной характеристики

$$S = \left| \frac{di_C}{dU_{ЗИ}} \right|.$$

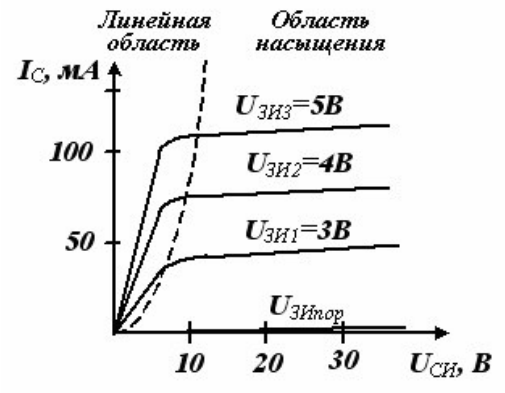



Рисунок 6. Выходные характеристики полевого транзистора КП902

Упрощенная схема усилительного каскада на полевом транзисторе с общим истоком показана на рисунке 7.

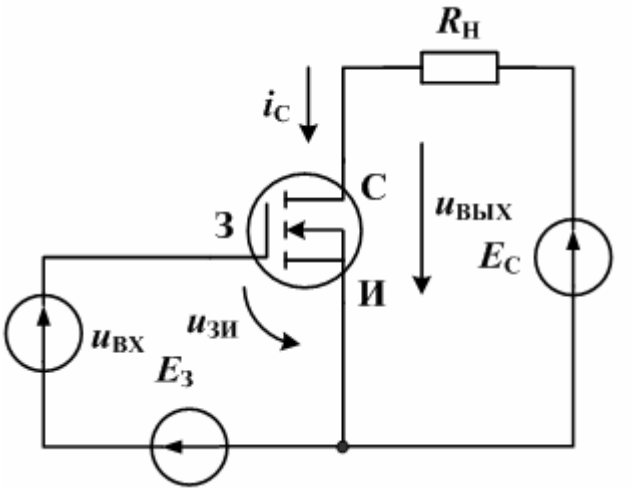



Рисунок 7. Схема усилителя на полевом транзисторе

Источник напряжения $E_З$ создает требуемое напряжение смещение на затворе. Источник напряжения $E_С$ – напряжение питания цепи стока. Источник переменного сигнала $u_С$ подключен между затвором и истоком. На рисунке 8 показана схема замещения усилительного каскада на полевом транзисторе в области низких частот для малых сигналов.

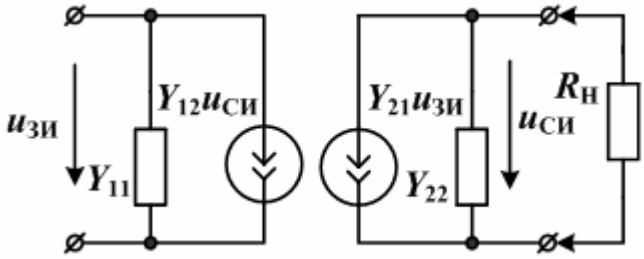



Рисунок 8. Схема замещения усилителя на полевом транзисторе в Y-параметрах

Переменное входное напряжение $u_{ЗИ}$ преобразуется управляемым источником тока в переменный ток стока $i_С$, который создает в нагрузке выходное напряжение $u_{СИ}$.

Этой схеме замещения (без учета нагрузки) соответствуют уравнения полевого транзистора в Y-параметрах:

$$\begin{aligned} i_з &= y_{11}u_{зи} + y_{12}u_{си}; \\ i_с &= y_{21}u_{зи} + y_{22}u_{си}. \end{aligned} \qquad (1)$$

В этих уравнениях $y_{11}$ – проводимость утечки затвора транзистора, $y_{22}$ – выходная проводимость, $y_{21} = S$ - крутизна полевого транзистора (или проводимость прямой передачи), $y_{12}$ - проводимость обратной передачи. Как правило, считают $y_{11} = y_{12} = 0$. Выходная проводимость

$$y_{22} = \frac{1}{R_{вых}},$$

причем $R_{вых}$ составляет 30 кОм и более.

Схема усилительного каскада с общим истоком, которая исследуется в лабораторной работе, показана на рисунке 9.

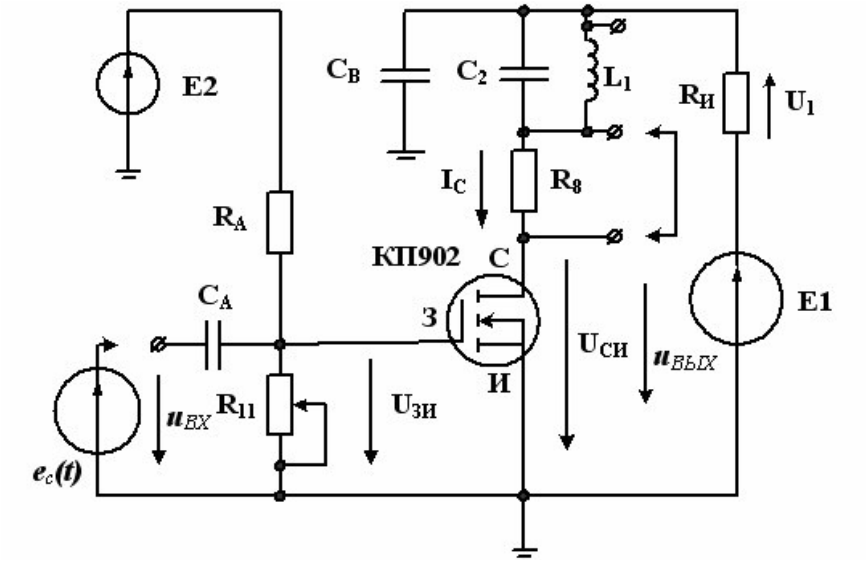



Рисунок 9. Схема усилительного каскада на полевом транзисторе

Источник постоянного напряжения $E_1$ создает режим по постоянному току в цепи стока. Источник $E_2$ создает смещение на затворе, которое регулируется потенциометром $R_{11}$. Усилительный каскад может работать как резистивный усилитель. В этом случае индуктивность $L_1$ следует замкнуть перемычкой и нагрузкой транзистора в цепи стока будет сопротивление $R_C = R_8 + R_И$. Для уменьшения искажений в резистивном усилителе напряжение смещения затвора выбирают таким, чтобы

$$U_{СИ} \approx \frac{E_1}{2}.$$

Входное сопротивление каскада без учета разделительной емкости $C_A$ равно

$$R_{BX} = \frac{R_A \cdot R_{11}}{R_A + R_{11}}. \quad (2)$$

Коэффициент усиления каскада по напряжению для малого переменного сигнала в режиме холостого хода без учета $C_A$ равен

$$\underline{K}'_{Ux} = -SR_C.$$

Выходное сопротивление полевого транзистора $R_{ВЫХ} \approx 30$ кОм много больше, чем $R_C = R_8 + R_И$ и не учитывается в резистивном усилителе.

С учетом коэффициента передачи входной цепи коэффициент усиления по напряжение составит

$$K_{Ux} = SR_C K_{ВЦ} = SR_C \frac{R_{BX} \cdot \omega C_A}{\sqrt{1 + (R_{BX} \cdot \omega C_A)^2}}. \quad (3)$$

В резонансном усилителе на полевом транзисторе в цепи стока включен параллельный колебательный контур, образованный индуктивностью $L_1$ и емкостью $C_2$. Конденсатор $C_B$ является блокировочным и шунтирует на высокой частоте источник напряжения $E_1$ и измерительный резистор $R_И$.

В резонансном усилителе на полевом транзисторе на средних частотах нагрузкой усилительного каскада является параллельное соединение комплексного сопротивления параллельного контура и выходного сопротивления полевого транзистора

$$\underline{Z}_c = \frac{\underline{Z}_к R_{вых}}{\underline{Z}_к + R_{вых}}. \quad (4)$$

По схеме замещения полевого транзистора вычислим выходное напряжение на контуре:

$$u_{вых}(f) = \frac{\frac{-Su_{вх}R_{рез}}{R_{рез}+R_{вых}}}{\sqrt{1+Q_{экв}^2(\frac{f}{f_{рез}}-\frac{f_{рез}}{f})^2}}, \quad (5)$$

где $R_{рез} = \frac{\rho^2}{R_к}$ – резонансное сопротивление контура; $f_{рез} = \frac{1}{2\pi\sqrt{LC}}$ – резонансная частота контура; $\rho = \sqrt{\frac{L}{C}}$ – характеристическое сопротивление контура; $R_к$ – сопротивление потерь в контуре ($R_к \approx 10$ Ом); $Q = \frac{\rho}{R_к}$; $Q_{экв} = \frac{Q}{1+\frac{R_{рез}}{R_{вых}}}$ – эквивалентная добротность контура с учетом потерь в катушках и внутреннего сопротивления источника сигнала $R_{вых}$.

***Лабораторное задание***

В схеме усилительного каскада с общим истоком (см. рисунок 9), который исследуется в лабораторной работе, значения элементов: $R_И$ = 10 Ом, $R_8$ = 510 Ом, $R_{11}$ = 2,2 кОм, $R_A$ = 2,2 кОм, $C_A = C_B$ = 2,2 мкФ. Параллельный колебательный контур образован индуктивностью $L_1$ = 10 мГн и емкостью $C_2$ = 68 нФ. По указанию преподавателя индуктивность и емкость могут быть с другими номиналами. Перемычкой нагрузку усилителя можно сделать резистивной (замкнута $L_1$) или резонансной (замкнуто $R_8$). Мультиметрами измеряются постоянные напряжения $U_{ЗИ}$, $U_{СИ}$ и напряжение $U_1$ на измерительном сопротивлении $R_И$ = 10 Ом.

Переменные сигналы измеряются двухканальным осциллографом.

А. Исследование передаточной характеристики полевого транзистора в схеме с общим истоком и определение крутизны вольтамперной характеристики.

1. При выполнении работы на стенде МЭЛ собрать схему (см. рисунок 9). Установить $E_1$ = 10 В, $E_2$ = 10 В, $R_{11}$ в крайнее левое положение, $e_c(t)$ не подключать, $R_8$ закоротить перемычкой.

2. Увеличивая $R_{11}$, измерять мультиметрами напряжения $U_{ЗИ}$ и $U_1$. Для каждого измерения рассчитать $I_C = \frac{U_1}{R_и}$.

Результаты записать в таблицу 1.

Таблица 1

| $U_{ЗИ}$, В | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| $U_{ЗИ(комп.)}$, В | - 6 | - 5 | - 4 | - 3 | - 2 | - 1 |
| $U_1$, В | | | | | | |
| $I_C$, мА | | | | | | |

Построить график передаточной характеристики.

В. Исследование выходных характеристик полевого транзистора в схеме с общим истоком.

1. В схеме измерений (см. рисунок 6) установить $E_2$ = 10 В, $U_{ЗИ}$ = 1 В. Входной сигнал $e_c(t)$ не подключать, $R_8$ закоротить перемычкой.

Изменяя $E_1$, провести измерения зависимости тока стока $I_C$ от напряжения сток-исток $U_{СИ}$. Результаты записать в таблицу 2.

Таблица 2

| | | $U_{СИ}$, В | 0 | 2 | 4 | 6 | 8 | 10 |
|---|---|---|---|---|---|---|---|---|
| $U_{ЗИ}$, В | 1 | $I_C$, мА | | | | | | |
| | 2 | $I_C$, мА | | | | | | |
| | 3 | $I_C$, мА | | | | | | |
| | 4 | $I_C$, мА | | | | | | |
| | 5 | $I_C$, мА | | | | | | |

2. Выполнить измерения для других значений напряжения затвор-исток, указанных в таблице 2.
3. По данным таблицы 2 построить семейство выходных характеристик полевого транзистора.

C. Исследование работы резистивного транзисторного усилителя с общим истоком в режиме малого сигнала.

1. В схеме (см. рисунок 9) включить $R_8$, индуктивность $L_1$ закоротить перемычкой.
2. Установить $E_1 = 10$ В. Регулируя $R_{11}$, установить $U_{СИ} = 5$ В.
3. Установить в функциональном генераторе частоту синусоидального сигнала 0,2 кГц, амплитуду входного сигнала $u_{ВХ} = 200$ мВ установить по осциллографу. Подключить входной сигнал к транзисторному усилителю.
4. Осциллографом наблюдать сигнал на выходе усилителя в режиме холостого хода без подключенной нагрузки. Если выходной сигнал не имеет существенных отличий от синусоидальной формы, измерить осциллографом амплитуду выходного сигнала. Если форма выходного сигнала существенно искажена, уменьшить амплитуду входного сигнала до 100 мВ. Записать измеренное значение $u_{ВЫХ}$. Рассчитать коэффициент усиления по напряжению

$$K_{Ux} = \frac{U_{ВЫХ}}{U_{ВХ}}.$$

5. Снять амплитудно-частотную характеристику транзисторного усилителя в режиме усиления малого сигнала, изменяя частоту входного сигнала в диапазоне от 200 Гц до 20 кГц. Результаты записать в таблицу 3.

Таблица 3

$U_{ВХ}$ = мВ

| F, кГц | 0,2 | 2 | 5 | 10 | 15 | 20 |
|---|---|---|---|---|---|---|
| $U_{ВЫХ}$ | | | | | | |
| $K_U(f)$ | | | | | | |

D. Исследование работы резонансного транзисторного усилителя с общим истоком в режиме малого сигнала.

1. В схеме (см. рисунок 8) разомкнуть индуктивность $L_1$ и закоротить сопротивление $R_8$.
2. Повторить исследования АЧХ-усилителя по п. С-5 для значений частоты 5, 6, 7, 8, 9 и 10 кГц. Результаты записать в таблицу 4, аналогичную таблице 3. Определить коэффициент усиления на резонансной частоте.
3. По данным таблицы 4 построить график амплитудно-частотной характеристики резонансного транзисторного усилителя. По графику АЧХ определить резонансную частоту и полосу пропускания усилителя.

E. Исследование искажений выходного сигнала в резистивном усилителе.

1. Установить $E_1$ = 10 В. Регулируя $R_{11}$, установить $U_{СИ}$ = 5 В. Установить частоту входного сигнала 20 кГц, напряжение 200 мВ. Наблюдая форму выходного сигнала, увеличить амплитуду входного сигнала до появления заметных искажений выходного сигнала. Зарисовать осциллограмму выходного сигнала и записать значение напряжения входного сигнала $u_{ВХ\ МАХ}$.

2. Установить частоту и амплитуду входного сигнала по п. С-3. Изменяя сопротивление $R_{11}$, наблюдать появление искажений формы выходного сигнала. Зарисовать осциллограмму выходного сигнала. Отключить входной сигнал и измерить напряжение затвор-исток, ток стока и напряжение сток-исток.

**Лабораторная работа № 10. Исследование операционных усилителей в цепях постоянного и переменного токов**

***Продолжительность: 180 минут.***

***Дисциплина: «Электротехника, электроника и схемотехника». Юнита 6.***

Предназначено для обучающихся по направлению «Информатика и ВТ» в соответствии с учебным планом.

**1 Вводная часть**

***Цель работы.*** Изучение типовых функциональных схем включения операционных усилителей и исследование их свойств.

Работа выполняется реальным моделированием на универсальном лабораторном стенде МЭЛ.

1. Перед выполнением лабораторной работы необходимо внимательно изучить правила техники безопасности, получить от преподавателя инструктаж по этим правилам и правилам поведения при выполнении лабораторной работы. В дальнейшем строго соблюдать правила техники безопасности и поведения в учебной лаборатории.

2. Перед выполнением лабораторной работы обучающемуся следует заранее изучить рекомендованный к данной теме теоретический материал, ознакомиться с описанием работы, продумать ответы на вопросы для самопроверки, подготовить в рабочем отчете бланк для заполнения протокола наблюдений. Бланк протокола наблюдений должен содержать наименование работы, схемы и таблицы для записи опытных данных. Лабораторные работы выполняются отдельными бригадами из двух-трех человек. Допускается иметь один рабочий отчет на бригаду. Рабочие отчеты должны оформляться в отдельной тетради для всего цикла лабораторных работ.

3. В начале лабораторной работы преподаватель проводит опрос обучающихся, проверяет наличие протоколов и готовность к работе.

4. Включение и выключение лабораторного стенда МЭЛ можно производить после допуска к работе.

5. Работа считается выполненной после утверждения преподавателем рабочего отчета бригады.

6. Для защиты лабораторной работы каждый обучающийся по каждой работе составляет индивидуальный отчет, который должен содержать:

- ✓ заглавие (номер и название лабораторной работы);
- ✓ схемы исследованных электрических цепей;
- ✓ результаты исследований (в виде таблиц и графиков);
- ✓ расчетную часть задания;
- ✓ выводы по работе.

7. Работа считается защищенной после собеседования, утверждения индивидуального отчета преподавателем и решения контрольного задания по работе.

*Описание набора элементов и приборов*

Элементы, используемые в данной лабораторной работе, размещены на второй панели МЭЛ (рисунок 1).

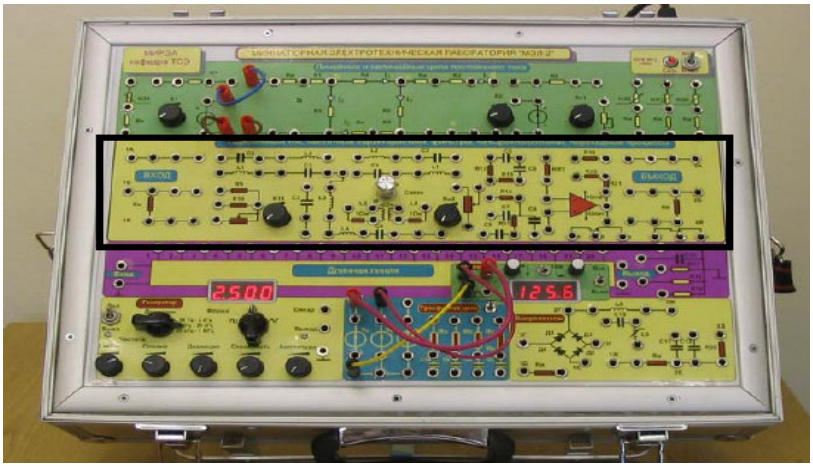
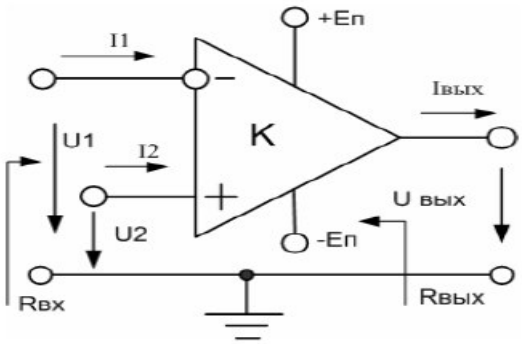


Рисунок 1. Панель МЭЛ для исследование операционных усилителей в цепях постоянного и переменного токов

**2 Теоретические сведения и методы расчета электрических цепей с операционными усилителями**

Операционные усилители (ОУ) выполняются в виде интегральных микросхем и имеют в полосе частот от 0 до десятков килогерц (современные ОУ до сотен мегагерц) собственный коэффициент усиления $K'_U$ – не менее нескольких тысяч. Обозначение ОУ на схемах показано на рисунке 2.




$$K'_U = \infty,\ \underline{U}_2 - \underline{U}_1 = 0,$$
$$I_{вх1} = 0, I_{вх2} = 0,$$
$$R'_{вх} = \infty,\ R'_{вых} = 0$$

Рисунок 2. Идеальный ОУ

Вход 1, обозначенный знаком (-), называют инвертирующим. Вход 2, обозначенный знаком (+), называют неинвертирующим. Входы питания +Еп и –Еп на схемах электрических цепей часто не обозначают. Выходное напряжение $\underline{U}_{вых} = K'_U(\underline{U}_2 - \underline{U}_1)$ в области низких частот совпадает по фазе с разностью $\underline{U}_2 - \underline{U}_1$.

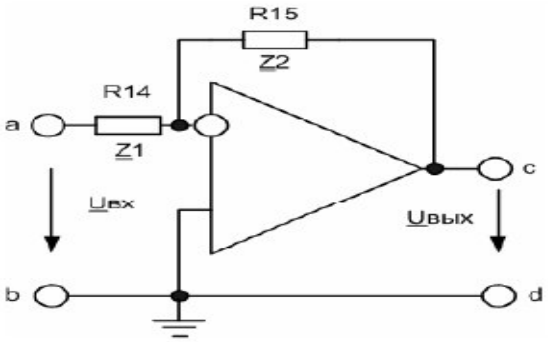
Поскольку $K'_U$ весьма велик, а $\underline{U}_{вых}$ ограничено $\left(\left|\underline{U}_{вых}\right| < \left|E_П\right|\right)$, то разность $\underline{U}_2 - \underline{U}_1 = U_\gamma \to 0$. Эту разность называют дифференциальным входным напряжением или виртуальным нулем.

Свойства идеального ОУ без внешних обратных связей используют при расчетах схем с ОУ. Технические характеристики реальных ОУ отличаются от идеальных.

Дифференциальный коэффициент усиления $K'_U = \dfrac{\Delta U_{вых}}{\Delta(U_2 - U_1)}$ имеет конечную величину, которая лежит в пределах $10^3$-$10^5$.

Входное сопротивление $R'_{вх}$ лежит в пределах от единиц до сотен Ом. Частота единичного усиления – от сотен килогерц до сотен мегагерц. Скорость нарастания напряжения (В/мкс), определяющая переходные характеристики ОУ, может составлять от десятков мВ/мкс до сотен В/мкс. Внешние элементы образуют внешние обратные связи (ОС) операционного усилителя. ОУ с ОС имеет передаточную функцию, которая определяется параметрами элементов и схемой включения ОС. При соответствующем выборе ОС операционный усилитель можно использовать для сложения, вычитания, дифференцирования, интегрирования сигналов, а также для получения различных функциональных зависимостей. За способность выполнять различные математические функции ОУ и получил название «операционный усилитель».

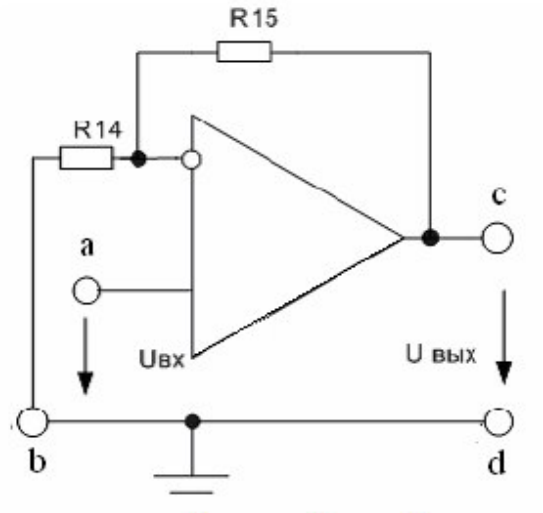
На следующих рисунках 3–6 показаны схемы с ОУ, которые исследуются в лабораторной работе, и основные расчетные формулы.




$$K'_U = \frac{\underline{U}_{вых}}{\underline{U}_{вх}} = -\frac{R_{15}}{R_{14}},$$

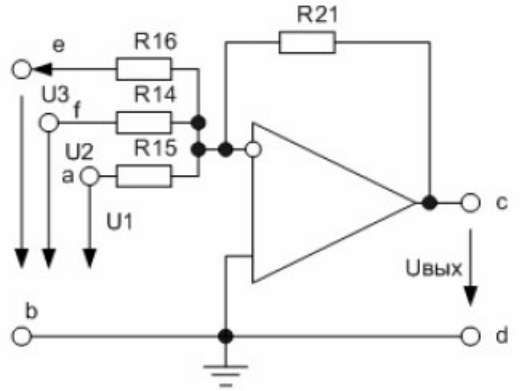
$$K'_U(p) = \frac{U_{вых}(p)}{U_{вх}(p)} = -\frac{Z_2(p)}{Z_1(p)}$$

Рисунок 3. Инвертирующий ОУ



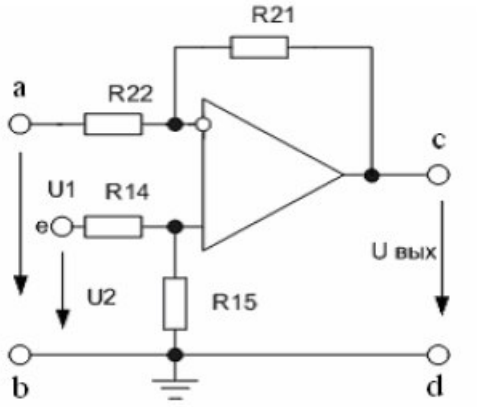

$$K'_U = \frac{U_{вых}}{U_{вх}} = \frac{R_{14} + R_{15}}{R_{14}}$$

Рисунок 4. Неинвертирующий ОУ




$$U_{вых} = -\left(\frac{R_{21}}{R_{15}}U_1 + \frac{R_{21}}{R_{14}}U_2 + \frac{R_{21}}{R_{16}}U_3\right)$$

Рисунок 5. Инвертирующий сумматор




$$U_{вых} = \left[ \left( 1 + \frac{R_{21}}{R_{22}} \right) / \left( 1 + \frac{R_{15}}{R_{14}} \right) \right] \cdot \frac{R_{15}}{R_{14}} U_2 - \frac{R_{21}}{R_{22}} U_1 .$$

При равных $R_{14} = R_{15} = R_{21} = R_{22}$

$$U_{вых} = U_2 - U_1$$

Рисунок 6. Вычитающий ОУ

*Частотно-зависимые звенья с операционными усилителями*

Простейшие частотно-зависимые звенья с операционными усилителями показаны на следующих рисунках 7–12.

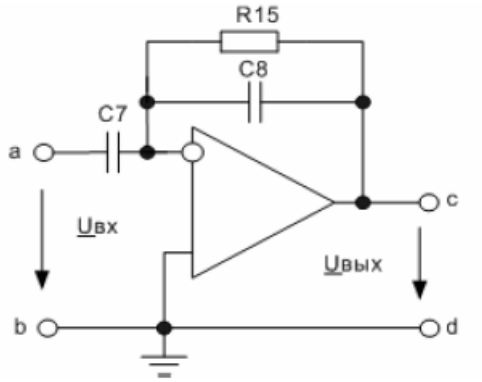



Рисунок 7. Дифференцирующее звено с коррекцией

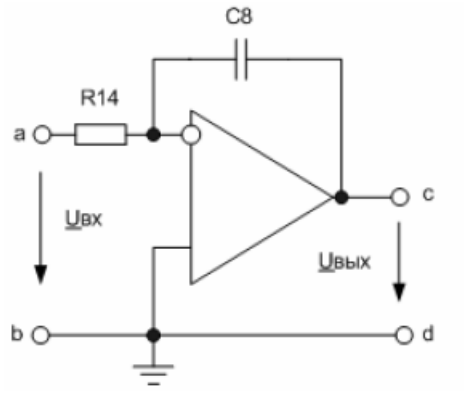



Рисунок 8. Интегрирующее звено

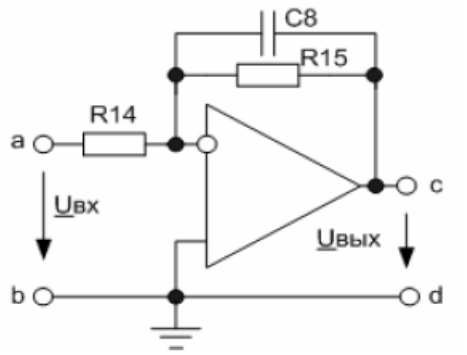



Рисунок 9. Активный фильтр низких частот первого порядка

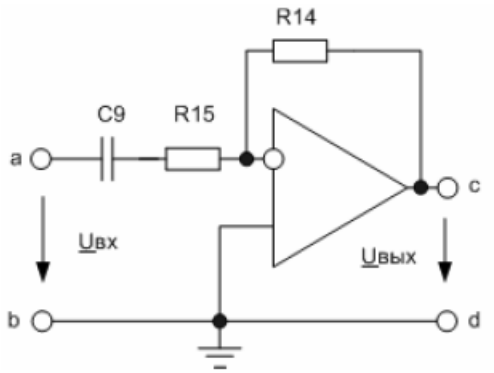
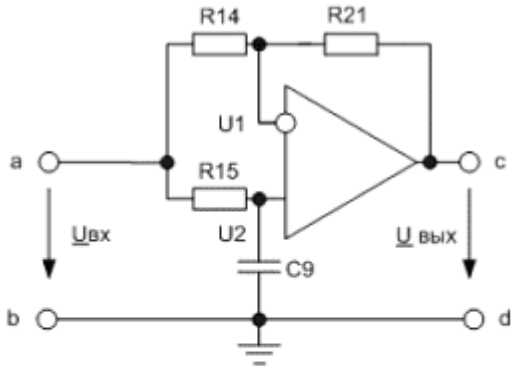



Рисунок 10. Активный фильтр высоких частот первого порядка




$$\underline{K}'(j\omega) = \frac{1 - j\omega RC_9}{1 + j\omega RC_9},$$

$$\left|\underline{K}'(j\omega)\right| = 1,$$

$$\varphi = -2 \operatorname{arctg}(\omega RC_9)$$

Рисунок 11. Фазовращатель

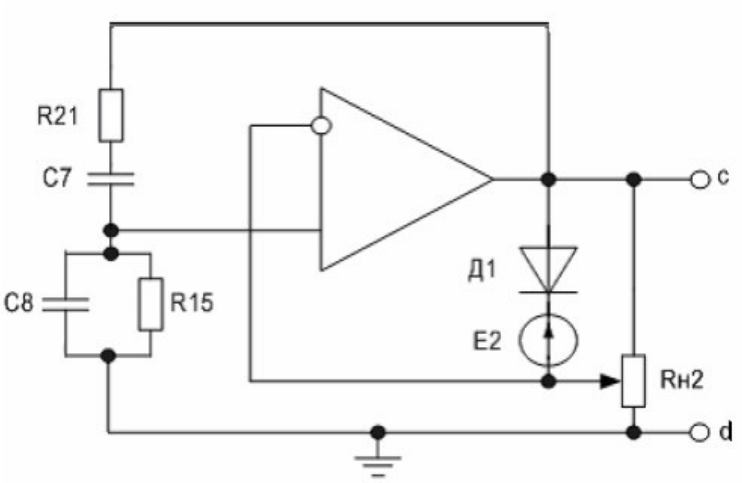



Рисунок 12. Генератор Вина

Четыре первые схемы соответствуют обобщенному инвертирующему усилителю (см. рисунок 3) с операторным коэффициентом передачи

$$K'_U(p) = \frac{U_{вых}(p)}{U_{вх}(p)} = -\frac{Z_2(p)}{Z_1(p)} \quad (1)$$

Подставляя в (1) комплексные сопротивления $\underline{Z}_1(j\omega)$ и $\underline{Z}_2(j\omega)$, можно вычислить амплитудно-частотную характеристику звена $(|\underline{K'}_U(j\omega)|)$ и фазочастотную характеристику звена $Arg(\underline{K'}_U(j\omega))$. Так, например, для активного ФНЧ первого порядка (см. рисунок 9) АЧХ будет определяться формулой

$$K'_U(\omega) = \frac{K}{\sqrt{1+(\frac{\omega}{\omega_p})^2}}, \quad (2)$$

где $K = \frac{R_{15}}{R_{14}}$, $\omega_p = \frac{1}{R_{15}C_8}$.

Для фазовращателя (см. рисунок 11) составим уравнения с учетом свойств идеального ОУ:

$$U_1(p) = \frac{(U_{вх}(p) - U_{вых}(p))R_{21}}{R_{14}+R_{21}} = U_2(p) = \frac{U_{вх}(p)\cdot\frac{1}{pC_9}}{R_{14}+\frac{1}{pC_9}}.$$

Отсюда при равных резисторах получим

$$\frac{U_{вх}(p)-U_{вых}(p)}{2U_{вх}(p)} = \frac{1}{1+pRC_9} \text{ и } \frac{U_{вых}(p)}{U_{вх}(p)} = \frac{1-pRC_9}{1+pRC_9}. \quad (3)$$

Для комплексной частотной характеристики $\underline{K}'(j\omega) = \frac{1-j\omega RC_9}{1+j\omega RC_9}$ модуль является постоянным и равным 1, а фаза $\varphi = -2\ \mathrm{arctg}(\omega RC_9)$.

Изменяя сопротивление R или емкость $C_9$, можно установить фазовый сдвиг в диапазоне от $0^o$ до $-180^o$.

В генераторе Вина (см. рисунок 12) при равных сопротивлениях $R_{21} = R_{15} = R$ и емкостях $C_7 = C_8$ на квазирезонансной частоте $f_k = \frac{1}{2\pi RC_8}$ фазовый сдвиг в цепи обратной связи равен нулю, а петлевой коэффициент передачи β=1/3. Для самовозбуждения коэффициент усиления усилителя К должен быть

больше 3. Это достигается регулировкой резистора $R_{н2}$. Диод и источник напряжения $E_2$ требуются для стабилизации режима генерации.

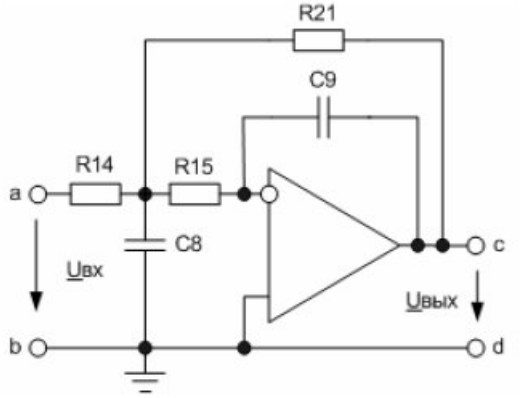
***Активные фильтры второго порядка***

Активные фильтры второго порядка содержат один или несколько ОУ с частотно-зависимыми обратными связями. В цепях ОС применяют резисторы и конденсаторы. Поэтому такие фильтры называют активные RC-цепи (ARC-цепи). Активные фильтры позволяют получить разнообразные частотные характеристики коэффициента передачи в диапазоне частот от нуля до нескольких мегагерц. Они более компактны и технологичны по сравнению с LC-фильтрами, не требуют применения индуктивностей.

Порядок активного фильтра определяется наибольшей степенью переменной p в знаменателе его передаточной функции. Фильтры высокого порядка имеют лучшие частотные характеристики.

Передаточные функции активных фильтров рассчитывают по уравнениям для операторной схемы замещения с учетом свойств операционного усилителя. Комплексную частотную характеристику получают заменой p на jω в передаточной функции. Амплитудно-частотные характеристики равны модулю комплексной частотной характеристики.

В лабораторной работе исследуются активные фильтры второго порядка с одним ОУ. Схемы фильтров и расчетные формулы показаны на следующих рисунках 13–16.

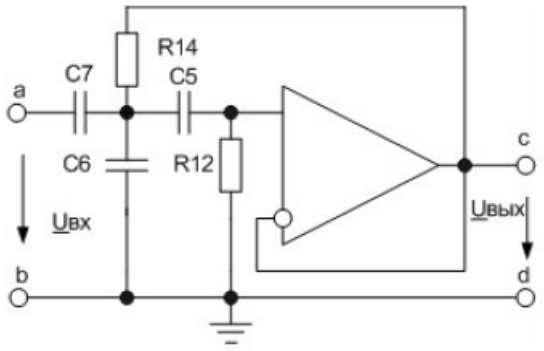



$$K(\omega)=\frac{K\omega_0^2}{\sqrt{(\omega^2-\omega_0^2)^2+\omega^2(\omega_0/q)^2}},$$

$$K=R_{21}/R_{14},\ \omega_0^2=\frac{1}{R_{15}\cdot R_{21}\cdot C_8\cdot C_9},$$

$$q=\sqrt{\frac{R_{15}\cdot C_8}{R_{21}\cdot C_9}}\cdot\frac{1}{1+R_{15}/R_{14}}$$

Рисунок 13. Активный фильтр низких частот

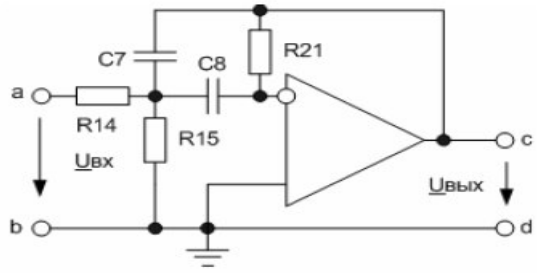



$$K(\omega)=\frac{K\omega^2}{\sqrt{(\omega^2-\omega_0^2)^2+\omega^2(\omega_0/q)^2}},$$

$$K=\frac{C_7}{C_7+C_6};\omega_0^2=\frac{1}{R_{14}\cdot R_{12}\cdot C_0\cdot C_5};$$

$$C_0=C_6+C_7;q=\frac{1}{(1+\frac{C_0}{C_5})}\sqrt{\frac{R_{12}\cdot C_0}{R_{14}\cdot C_5}}$$

Рисунок 14. Активный фильтр высоких частот



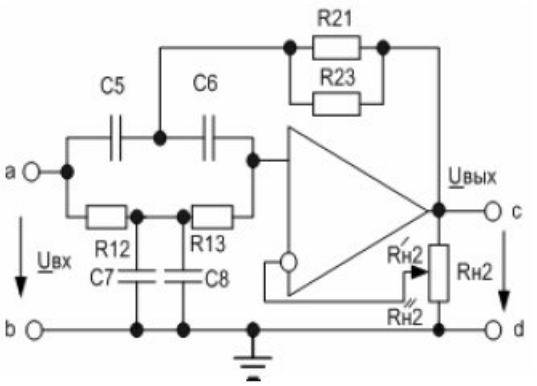

$$K(\omega)=\frac{K\omega(\omega_0/q)}{\sqrt{(\omega^2-\omega_0^2)^2+\omega^2(\omega_0/q)^2}},$$

$$K=\frac{R_{15}K_0}{R_{15}+R_{14}};K_0=q^2(1+C_8/C_7);$$

$$R_0=\frac{R_{14}\cdot R_{15}}{R_{14}+R_{15}};\omega_0^2=\frac{1}{R_0\cdot R_{21}\cdot C_7\cdot C_8};$$

$$q=\sqrt{\frac{R_{21}\cdot C_7}{R_0\cdot C_8}}\cdot\frac{1}{1+C_7/C_8}$$

Рисунок 15. Активный полосовой фильтр




$$K(\omega)=\frac{K\left|\omega^2-\omega_0^2\right|}{\sqrt{(\omega^2-\omega_0^2)^2+4\omega^2\omega_0^2(2-K)^2}},$$

$$\omega_0=\frac{1}{R_{12}C_5},K=1+\frac{R'_{H2}}{R''_{H2}}$$

Рисунок 16. Активный заграждающий фильтр

***Лабораторное задание***

Схема измерений показана на рисунке 17.

В ней используются функциональный генератор ГС, управляемые источники напряжений $E_1$ и $E_2$, один из генераторов трехфазного напряжения $e_a$, осциллограф, вольтметр переменного тока V, мультиметры. Эта схема и набор элементов применяется во всех лабораторных работах с ОУ.

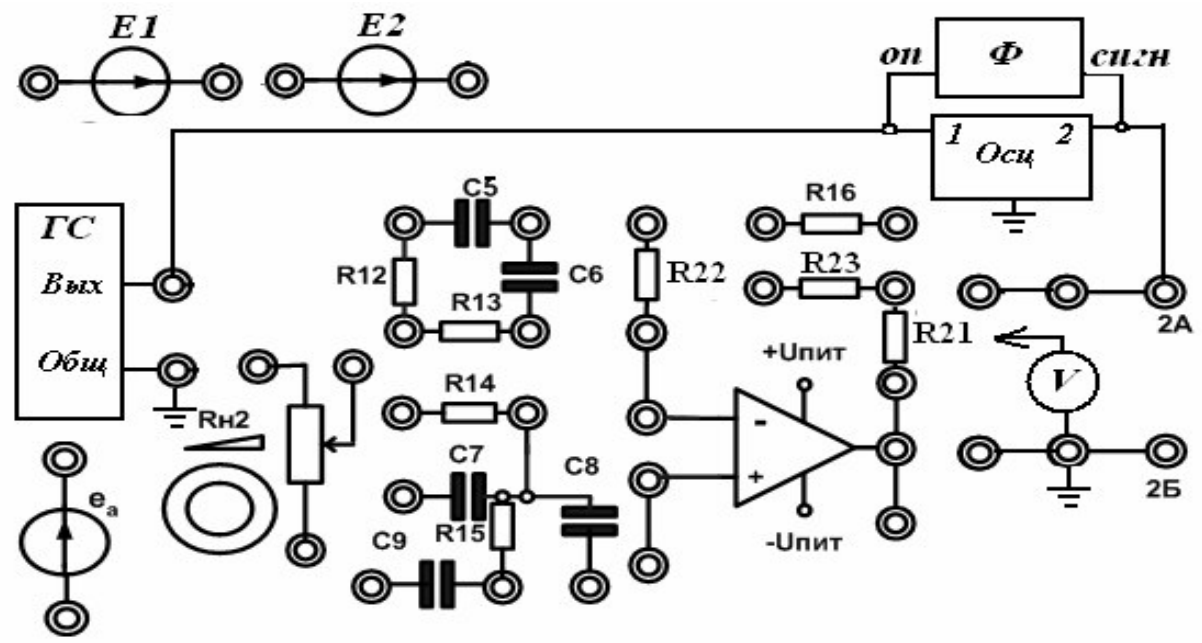



Рисунок 17. Схема измерений

1. Собрать схему инвертирующего ОУ (см. рисунок 3). Узлы b, d, неинвертирующий вход ОУ, общие клеммы ГС и осциллографа соединить с клеммами 2Б. Выход ОУ (узел *c*) соединить с клеммами 2А. Установить напряжение $E_2$ равным 2 В и подключить к узлам *a*, *b*. Мультиметром измерить напряжение на выходе. Записать результаты. Отключить $E_2$.

2. Установить на ГС гармонический сигнал с напряжением 100 мВ, частотой 100 Гц и подключить к узлу *a*. Высокочастотным вольтметром измерить выходное напряжение ОУ. Осциллографом наблюдать инверсию фазы. Изменяя частоту сигнала в диапазоне от 100 Гц до 100 кГц, снять амплитудно-частотную характеристику ОУ и определить полосу пропускания по уровню -3 дб. Результаты записать в таблицу 1.

Таблица 1

| | | | | | |
|---|---|---|---|---|---|
| Инвертирующий ОУ | *f*, кГц | 0,1 | | | 100 |
| | $U_{вых}$ | | | | |
| | $K_U$ | | | | |
| Неинвертирую-щий ОУ | *f*, кГц | | | | |
| | $U_{вых}$ | | | | |
| | $K_U$ | | | | |

3. Увеличить напряжение входного сигнала до 1 В. Увеличивая частоту от 100 Гц, оценить по осциллографу предельную частоту $f_{пр}$, при которой начинается искажение гармонического сигнала, вызванное ограниченной скоростью нарастания напряжения в ОУ.

4. Собрать схему неинвертирующего ОУ (см. рисунок 4). Повторить измерения по п. 1 и 2.

5. Собрать схему инвертирующего сумматора (см. рисунок 5). На вход *e* подключить гармонический источник $e_a$ генератора трехфазной цепи. На вход *f* подключить от ГС гармонический сигнал с амплитудой 200 мВ и частотой 500 Гц. На вход *a* включить постоянное напряжение 3 В от источника напряжения $E_2$. Зарисовать осциллограммы выходного сигнала.

6. Собрать схему вычитающего ОУ (см. рисунок 6). Установить напряжение источника $E_1 = 3$ В и подключить к узлу *a*. Установить напряжение источника $E_2 = 2$ В и подключить к узлу *e*. Измерить мультиметром выходное напряжение ОУ.

7. Подключить к узлу *a* вместо $E_1$ гармонический сигнал от ГС с частотой 100 Гц и амплитудой 500 мВ. Наблюдать выходной сигнал осциллографом. Регулируя $E_2$, выполнить смещение выходного сигнала в положительную область до начала амплитудного ограничения гармонической составляющей. Определить динамический диапазон ОУ. Зарисовать осциллограммы для исходных сигналов и в случае ограничения гармонического сигнала.

8. По экспериментальным данным построить амплитудно-частотные характеристики инвертирующего ОУ и неинвертирующего ОУ.

9. Представить результаты измерений преподавателю и после его проверки и одобрения выключить приборы и стенд.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРОФЕССИОНАЛЬНЫХ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (ПЛЗ)**
**ПО ДИСЦИПЛИНЕ «ЭЛЕКТРОТЕХНИКА, ЭЛЕКТРОНИКА И СХЕМОТЕХНИКА»**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки В.Г. Буцкая

**4924.01.01;МУАТ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

по учебной дисциплине «Электротехника, электроника и схемотехника»

по разделу «Электрические цепи при постоянных и синусоидальных токах и напряжениях»

# КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ (РЕШЕНИЕ ЗАДАЧ)

**МОСКВА 2018**

Разработано Д.П. Гуриным

Под ред. Э.М. Берлинера, д.т.н., проф.

Рекомендовано Учебно-методическим советом в
качестве методического пособия для обучающихся

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

по учебной дисциплине «Электротехника, электроника и схемотехника»
по разделу «Электрические цепи при постоянных и синусоидальных токах и напряжениях»

Методические указания подготовлены для обучающихся в образовательной организации по проведению коллективного тренинга по дисциплине «Электротехника, электроника и схемотехника» по направлению 09.03.01 «Информатика и вычислительная техника». МУ являются неотъемлемой частью дидактического обеспечения проведения практических занятий и нацелены на формирование общекультурных, общепрофессиональных, профессиональных или профессионально-прикладных компетенций у обучающихся.

**С О Д Е Р Ж А Н И Е**

Стр.

**I ВВЕДЕНИЕ**

- **Модуль:** «Электрические цепи при постоянных и синусоидальных токах и напряжениях».
- **Цель семинара** – формирование у бакалавров целостного представления об электротехнике как об инструменте, позволяющем анализировать и решать теоретические и практические задачи, связанные с их будущей профессиональной деятельностью.
- **Задачи семинара:**

познакомить обучающихся с методологией изучаемой дисциплины;

способствовать формированию базы научных знаний по электротехнике;

развить у обучающихся творческий подход к теоретическому материалу, физическим трактовкам явлений и процессов, происходящих в электрических цепях;

развитие навыков самостоятельной работы;

повышение мотивации к процессу изучения учебной дисциплины.

**II УЧЕБНО-МЕТОДИЧЕСКОЕ, МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ**

**а) Литература**

**Основная учебная**

1. **Лаппи Ф.Э.** Минимальный курс электротехники и электроники. Часть 1. Основные элементы электротехники и электроники [Электронный ресурс]: учебное пособие/ Лаппи Ф.Э.— Электрон. текстовые данные.— Новосибирск: Новосибирский государственный технический университет, 2014.— 112 c.— http://www.iprbookshop.ru/45112.— ЭБС «IPRbooks»

2. **Трубникова В.Н.** Электротехника и электроника. Часть 1. Электрические цепи [Электронный ресурс]: учебное пособие/ Трубникова В.Н.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2014.— 137 c. http://www.iprbookshop.ru/33672.— ЭБС «IPRbooks»

3. **Букштынович, И.М.** Микропроцессорные БИС/СБИС. Интерфейсные БИС/СБИС в микропроцессорных комплектах. [Электронный ресурс]: рабочий учебник/ Букштынович, И.М. - 2013. - http://lib.muh.ru.

4.. **Гурин, Д.П.** Электронные устройства и преобразователи. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

5. **Гурин, Д.П.** Электронные приборы. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

6. **Гурин, Д.П.** Трансформаторы, электрические машины, электроизмерительные приборы и электрические измерения. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2012. - http://lib.muh.ru.

**Дополнительная**

2. **Дураджи, В.Н.** Электрические цепи при несинусоидальных токах и напряжениях. Магнитные цепи. [Электронный ресурс]: рабочий учебник/ Дураджи, В.Н. - 2011. - http://lib.muh.ru.

2. **Гурин, Д.П.** Четырехполюсники. Электрические фильтры. Переходные процессы в линейных электрических цепях. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2011. - http://lib.muh.ru.

3. **Гурин, Д.П.** Электрические цепи при постоянных и синусоидальных токах и напряжениях. [Электронный ресурс]: рабочий учебник/ Гурин, Д.П. - 2011. - http://lib.muh.ru.

**б) Информационное обеспечение**

Ресурсы информационно-телекоммуникационной сети Интернет:

http://www.electrolibrary.info/
- http://www.e-scientist.ru/
- http://electrono.ru/.

Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы;
- тренинговые и тестирующие программы;
- интеллектуальные роботизированные системы оценки качества выполненных работ.

Роботизированные системы для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ИС «Комбат»;
- ИС «ЛиК»;
- ИР «КОП»;
- ИИС «Каскад».

**в) Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**III Организационно-методические указания и рекомендации**

При подготовке к коллективному алгоритмическому тренингу обучающийся обязан изучить учебники по предмету (рабочие учебники/юниты), просмотреть слайд-лекции (при наличии) по модулю на личном компьютере или в аудитории индивидуального компьютерного тренинга. Затем обучающийся должен изучить основную литературу по теме занятия и источники из списка дополнительной литературы, используемые для расширения объема знаний обучающегося.

Обучающийся имеет возможность изучить электронные учебные, учебно-методические и научные издания, доступные в ТКДБ.

Подготовка к коллективному алгоритмическому тренингу осуществляется обучающимся в рамках самостоятельной работы. При подготовке к занятиям необходимо ознакомиться с заданиями и разобрать алгоритм решения задач каждого типа. После освоения алгоритмов решения задач обучающемуся следует попробовать выполнить предложенные задания этого типа. При возникновении сложностей с решением задач обучающемуся следует обратиться к материалам рабочих учебников, слайд-лекций, основной и дополнительной литературы и к другим информационным образовательным ресурсам образовательной организации.

**IV Решение типовых задач**

**Тема 1 «Электрические цепи постоянного тока»**

**Задача 1**

На рисунке 1.1 показана схема электрической цепи с резисторами, сопротивления которых $R_1 = 18$ Ом, $R_2 = 30$ Ом, $R_3 = 20$ Ом.

Определить токи ветвей, если напряжение U = 120 В.

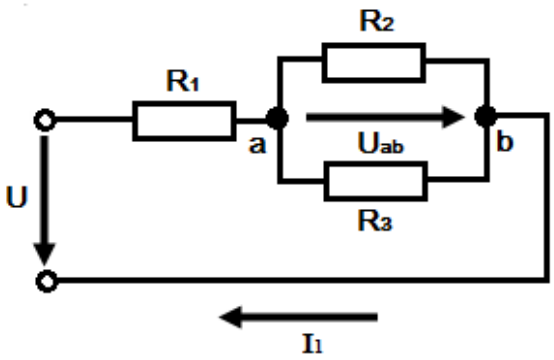



Рисунок 1.1. Схема цепи

**Решение**

Эквивалентное сопротивление разветвленного участка цепи

$$R_{23} = \frac{R_2 R_3}{R_2 + R_3} = \frac{30 \times 20}{30 + 20} = 12 \text{ Ом}.$$

Общее сопротивление цепи

$$R = R_1 + R_{23} = 18 + 12 = 30 \text{ Ом}.$$

В соответствии с законом Ома ток

$$I_1 = \frac{U}{R} = \frac{120}{30} = 4A.$$

Напряжение на зажимах параллельных ветвей

$$U_{ab} = R_{23} I_1 = 12 \cdot 4 = 48 \text{ В}.$$

Токи ветвей

$$I_2 = \frac{U_{ab}}{R_2} = \frac{48}{30} = 1{,}6\ A;$$

$$I_3 = \frac{U_{ab}}{R_3} = \frac{48}{20} = 2{,}4\ A.$$

**Задача 2**

Найти распределение токов в схеме цепи рисунка 1.2, если $R_1 = R_2 = 0{,}5$ Ом, $R_3 = 6$Ом; $R_4 = 6$ Ом; $R_5 = R_6 = 1$ Ом, $R_7 = 2$ Ом, напряжение на входе U = 120 В.

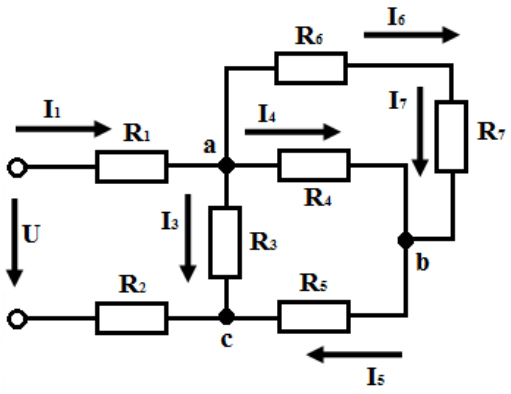



Рисунок 1.2. Схема цепи

**Решение**

Определяем общее сопротивление цепи, заменяя отдельные участки ее эквивалентными сопротивлениями:

$$R_{ab} = \frac{(R_6 + R_7) \times R_4}{R_6 + R_7 + R_4} = \frac{(1+2) \times 6}{1+2+6} = 2 \text{ Ом};$$

$$R_{ac} = \frac{(R_{ab} + R_5) \times R_3}{R_{ab} + R_5 + R_3} = \frac{(2+1) \times 6}{2+1+6} = 2 \text{ Ом}.$$

Общее или входное сопротивление цепи

$$R_{вх} = R_{ac} + R_1 + R_2 = 2 + 0{,}5 + 0{,}5 = 3 \text{ Ом}.$$

Ток в неразветвленной части цепи, т.е. на участках с резисторами $R_1$ и $R_2$:

$$I_1 = \frac{U}{R_{вх}} = \frac{120}{3} = 40A.$$

Напряжение $U_{ac}$ согласно второму закону Кирхгофа

$$U_{ac} = U - (R_1 + R_2) \cdot I_1 = 120 - (0{,}5 + 0{,}5) \cdot 40 = 80 \text{ В}.$$

В резисторе $R_3$ ток определяется по закону Ома:

$$I_3 = \frac{U_{ac}}{R_3} = \frac{80}{6} = 13{,}33A.$$

В резисторе $R_5$ ток

$$I_5 = \frac{U_{ac}}{(R_{ab} + R_5)} = \frac{80}{2+1} = 26{,}7A.$$

Напряжение на участке ab

$$U_{ab} = R_{ab} \cdot I_5 = 2 \cdot 26{,}7 = 53{,}4 \text{ В}.$$

В резисторе $R_4$ ток

$$I_4 = \frac{U_{ab}}{R_4} = \frac{53{,}4}{6} = 8{,}9A.$$

В резисторах $R_6$ и $R_7$ ток

$$I_6 = I_7 = \frac{U_{ab}}{R_6 + R_7} = \frac{53{,}4}{1+2} = 17{,}8A.$$

**Задача 3**

Определить токи в схеме рисунка 1.3, если $E_1 = 11$В, $E_2 = 2$ В, $R_1 = 1$ Ом, $R_2 = 2$ Ом, $J = 3$ А.

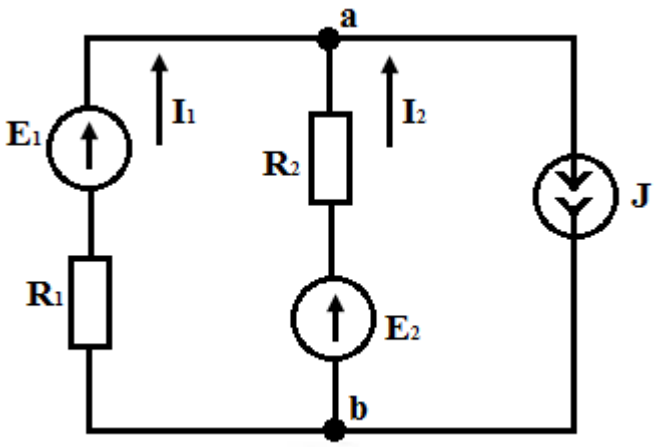



Рисунок 1.3. Схема цепи

***Решение***

С помощью формулы узлового напряжения находим

$$U_{ab} = \frac{(G_1 \times E_1 + G_2 \times E_2 - J)}{G_1 + G_2} = \frac{\frac{1}{1} \times 11 + \frac{1}{2} \times 2 - 3}{\frac{1}{1} + \frac{1}{2}} = 6\ B,$$

где $G_1 = \frac{1}{R_1}$; $G_2 = \frac{1}{R_2}$ – проводимости.

Токи $I_1$ и $I_2$ определяются по закону Ома для активного участка цепи

$$I_1 = \frac{E_1 - U_{ab}}{R_1} = \frac{11-6}{1} = 5\ A;$$

$$I_2 = \frac{E_2 - U_{ab}}{R_2} = \frac{2-6}{2} = -2\ A.$$

Ток $I_2$ направлен от точки *a* к точке *b*.

**Тема 2 «Электрические цепи однофазного синусоидального тока»**

**Задача 1**

В сеть напряжением $U$ = 220 В и частотой f = 50 Гц включаются поочередно реостат с сопротивлением 10 Ом (рисунок 2.1), индуктивная катушка с индуктивностью L = 32 мГн (рисунок 2.2) и конденсатор емкостью 317 мкФ (рисунок 2.3). Определить для каждого случая токи в приемниках, построить векторные диаграммы.

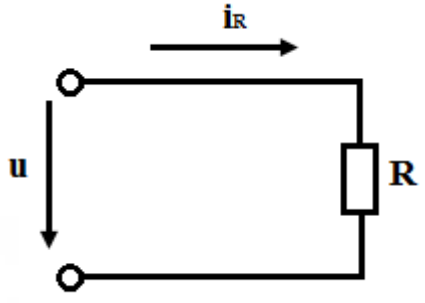



Рисунок 2.1. Схема с активным сопротивлением

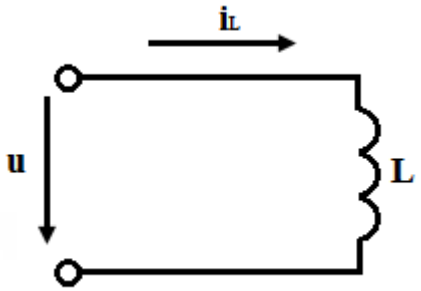



Рисунок 2.2. Схема с индуктивностью

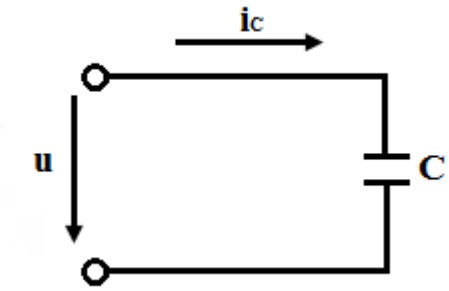



Рисунок 2.3. Схема с ёмкостью

**Решение**

Комплексные сопротивления:

$$\dot{Z}_R = 10 \text{ Ом};$$

$$\dot{Z}_L = jL\omega = jL2\pi f = j32 \times 10^{-3} \times 314 = j10 \text{ Ом};$$

$$\dot{Z}_C = -j\frac{1}{C\omega} = -j\frac{10^6}{317 \times 314} = -j10 \text{ Ом}.$$

Направление вектора $U$ на комплексной плоскости выбираем по оси +1, тогда $U = 220$ В.

Комплексные действующие значения токов:

$$\dot{I}_R = \frac{U}{R} = \frac{220}{10} = 22A;$$

$$\dot{I}_L = \frac{U}{jX_L} = \frac{220}{j10} = -j22 = 22e^{-\frac{j\pi}{2}} A;$$

$$\dot{I}_C = \frac{U}{-jX_C} = \frac{220}{-j10} = j22 = 22e^{\frac{j\pi}{2}} A.$$

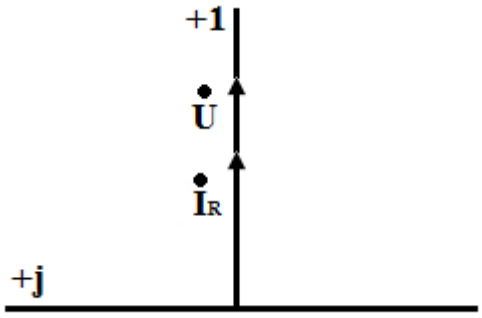
Мгновенные значения токов:

$$i_R = 22\sqrt{2}\sin\omega t \ A;$$

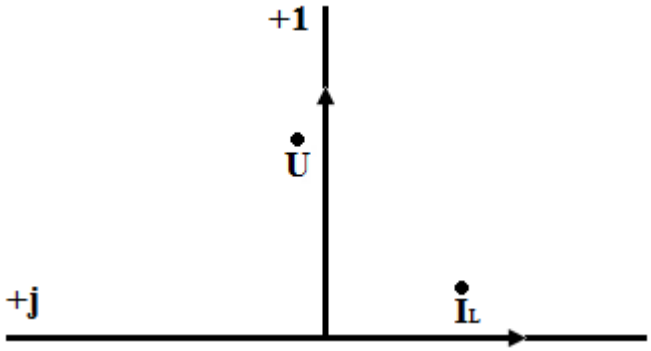
$$i_L = 22\sqrt{2}\sin(\omega t - \frac{\pi}{2}) \ A;$$

$$i_C = 22\sqrt{2}\sin(\omega t + \frac{\pi}{2})\ A\,.$$

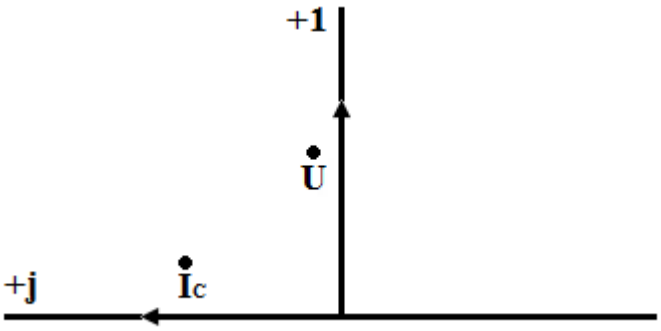
***Векторные диаграммы***




Векторная диаграмма для схемы с активным сопротивлением




Векторная диаграмма для схемы с индуктивностью




Векторная диаграмма для схемы с ёмкостью

**Задача 2**

В сеть напряжением U = 120 В и частотой f = 50 Гц включена индуктивная катушка сопротивлением R = 12 Ом и индуктивностью L = 66,2 мГн. Схема замещения сети изображена на рисунке 2.4. Определить комплексный ток, значения полной, активной и реактивной мощностей.

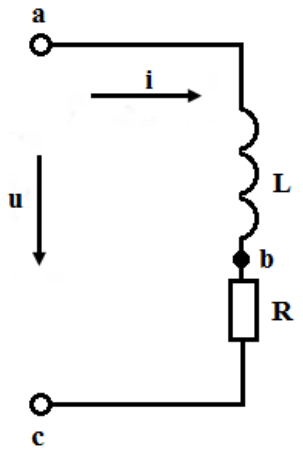



Рисунок 2.4. Схема замещения сети

**Решение**

Индуктивное сопротивление катушки

$$X_L = L\omega = 66{,}2\cdot10^{-3}\cdot314 = 20{,}8 \text{ Ом}.$$

Комплексное сопротивление

$$\dot{Z} = R + jX_L = 12 + j20{,}8 = 24e^{j60^\circ} \text{Ом}.$$

Комплексный ток

$$\dot{I} = \frac{U}{\dot{Z}} = \frac{120}{24e^{j60^\circ}} = 5e^{-j60^\circ} A.$$

Комплексная мощность:

$$\dot{S} = U \times \overset{*}{I} = 120 \times 5e^{j60^\circ} = 600e^{j60^\circ} = 600\cos 60^\circ + j600\sin 60^\circ = (300 + j520) \text{ ВА}.$$

Откуда P = 300 Вт, Q = 520 Вар, S = 600 ВА.

**Задача 3**

В сеть напряжением $U$ = 220 В и частотой f = 50 Гц включены последовательно соединенные батарея конденсаторов емкостью $C$ = 290 мкФ и резистор с сопротивлением $R$ = 5 Ом. Схема замещения сети изображена на рисунке 2.5.

Определить комплексный ток, полную, активную и реактивную мощности.

Комплексное сопротивление

$$\dot{Z} = R - j\frac{1}{C\omega} = 5 - j\frac{10^6}{290\times314} = 5 - j11 = 12{,}08e^{-j65{,}6^\circ} Ом.$$

Комплексные напряжение и ток:

$$\dot{U} = 220 \text{ В};$$

$$\dot{I} = \frac{\dot{U}}{\dot{Z}} = \frac{220}{12{,}08e^{-j65{,}6^\circ}} = 18{,}2e^{j65{,}6^\circ} A.$$

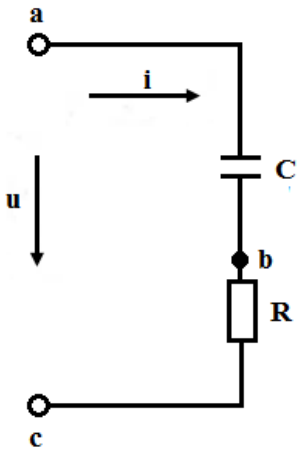



Рисунок 2.5. Схема замещения сети

Комплексная мощность

$$\dot{S} = \dot{U} \times \overset{*}{I} = 220 \times 18{,}2 e^{-j65{,}6^\circ} = 4006{,}6 e^{-j65{,}6^\circ} = (1657{,}9 - j3647{,}5)\ \text{ВА}.$$

Полная, активная и реактивная мощности: $S$ = 4006,6 ВА, $P$ = 1657,9 Вт, $Q_c$ = 3647,5 Вар.

**Задача 4**

В сеть (рисунок 2.6) напряжением 220 В включены последовательно катушка с активным сопротивлением 10 Ом и индуктивностью 159 мГн, а также батарея конденсаторов. Определить емкость батареи, при которой в цепи установится резонанс напряжений. Найти ток в цепи и напряжения на индуктивном и емкостном элементах.

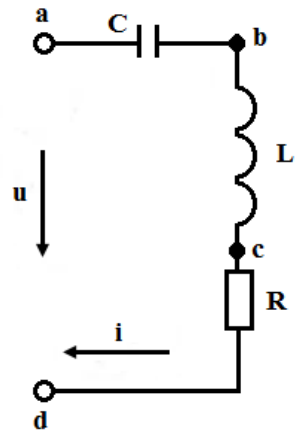



Рисунок 2.6. Схема замещения сети

**Решение**

Сопротивления реактивных элементов цепи при резонансе равны:

$$L \times \omega = \frac{1}{C_{рез} \times \omega};$$

$$C_{рез} = \frac{1}{L \times \omega^2} = 63{,}5\ \text{мкФ};$$

$$X_L = X_C = 50\ \text{Ом}.$$

Комплексное входное сопротивление схемы при резонансе будет чисто активным:

$$\dot{Z}_{вх} = R + jX_L - jX_C = R = 10 \text{ Ом}.$$

Ток

$$\dot{I}_{рез} = \frac{\dot{U}}{R} = 22 \text{ А}.$$

Напряжения на индуктивном и емкостном элементах равны между собой:

$$U_L = U_C = X \times I = 50 \times 22 = 1100 \text{ В}.$$

**Задача 5**

В сеть напряжением $U$ = 100 В включены резистор, индуктивная катушка и конденсатор (рисунок 2.7). Определить токи, если параметры цепи равны $R = X_C = 2X_L = 20$ кОм.

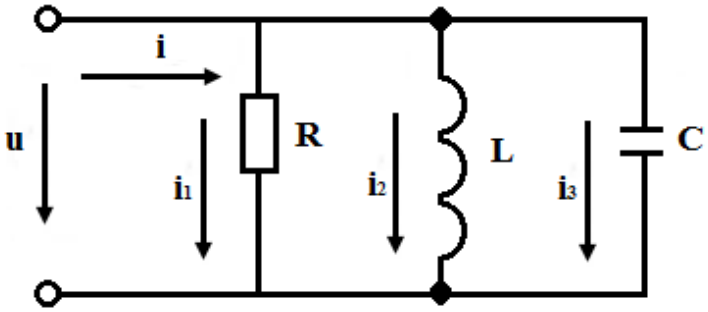



Рисунок 2.7. Схема замещения сети

**Решение**

Комплексные сопротивления ветвей:

$$\dot{Z}_1 = R = 20 \text{ кОм};$$

$$\dot{Z}_2 = jX_L = j10 \text{ кОм};$$

$$\dot{Z}_3 = -jX_C = -j20 \text{ кОм}.$$

Комплексные токи ветвей:

$$\dot{I}_1 = \frac{\dot{U}}{\dot{Z}_1} = 5 \text{ мА};$$

$$\dot{I}_2 = \frac{\dot{U}}{\dot{Z}_2} = -j10мА = 10e^{-j\frac{\pi}{2}} \text{ мА};$$

$$\dot{I}_3 = \frac{\dot{U}}{\dot{Z}_3} = j5мА = 5e^{j\frac{\pi}{2}} \text{ мА}.$$

Комплексный ток в неразветвленной части цепи

$$\dot{I} = \dot{I}_1 + \dot{I}_2 + \dot{I}_3 = 5 - j10 + j5 = 5 - j5 = 5\sqrt{2}e^{-j\frac{\pi}{4}} \text{ мА}.$$

**Задача 6**

Определить значение емкости С конденсатора, при котором в цепи рисунка 2.8, а установится резонанс токов. Найти входное сопротивление цепи при резонансе, а также токи ветвей. Напряжение сети U = 120 В, параметры цепи равны: $R = 3$ Ом, $X_L = 4$ Ом.

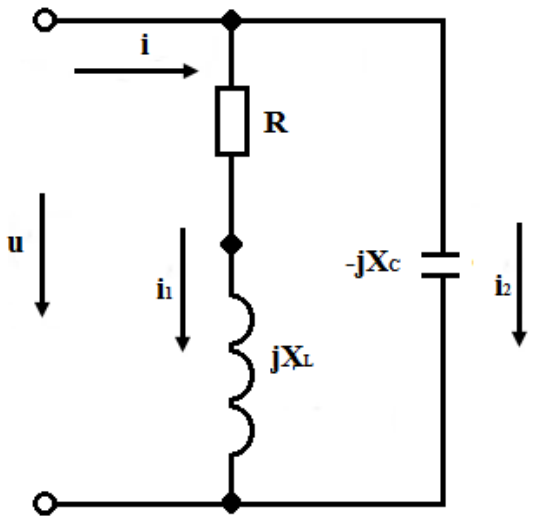



Рисунок 2.8. Схема замещения сети

Решение

Условием резонанса токов является равенство модулей реактивных проводимостей ветвей $B_L = B_C$.

Для рассматриваемой схемы:

$$B_L = \frac{X_L}{R^2 + X_L^2} = \frac{4}{25} = 0{,}16 \text{ См};$$

$$B_C = \omega C;$$

$$C = \frac{0{,}16}{314} \times 10^6 = 509{,}5 \text{ мкФ};$$

$$\dot{Z}_{вх} = \frac{\dot{Z}_1 \times \dot{Z}_2}{\dot{Z}_1 + \dot{Z}_2} = \frac{(R + jX_L)(-jX_C)}{R + jX_L - jX_C} \approx 8{,}37 \text{ Ом}.$$

Ток в неразветвленной части цепи

$$\dot{I} = \frac{\dot{U}}{\dot{Z}_{вх}} = \frac{120}{8{,}33} \approx 14{,}4 \text{ А}.$$

Токи ветвей:

$$\dot{I}_1 = \frac{\dot{U}}{\dot{Z}_1} = 24e^{-j53^\circ} = (14{,}4 - j19{,}2) \text{ А};$$

$$\dot{I}_2 = \frac{\dot{U}}{\dot{Z}_2} = 19e^{j90^\circ} = j19{,}2 \text{ А}.$$

**Тема 3 «Трехфазные электрические цепи»**

**Задача 1**

К трёхфазной сети с линейным напряжением 380 В (рисунок 3.1) подключена симметричная нагрузка, соединённая по схеме "звезда". Каждая из фаз представляет собой последовательно соединённые активное ($R_ф$ = 5 Ом) и индуктивное ($X_L$ = 8 Ом) сопротивления. Рассчитать активную и реактивную составляющие линейных токов, а также суммарную активную, реактивную и полную мощности.

Рисунок 3.1. Схема замещения цепи

**Решение**

Рассчитаем полное сопротивление для фазы:

$$Z_Ф = \sqrt{R_Ф^2 + X_Ф^2} = \sqrt{5^2 + 8^2} = 9{,}43 \text{ Ом}.$$

Определим фазное напряжение:

$$U_Ф = \frac{U_Л}{\sqrt{3}} = \frac{380}{\sqrt{3}} = 220 \text{ В}.$$

Определим ток в каждой фазе:

$$I_Ф = \frac{U_Ф}{Z_Ф}\frac{220}{9{,}43} = 23{,}3 \text{ А}.$$

Суммарная потребляемая активная мощность

$$P_\Sigma = 3P_Ф = 3I_Ф^2 R_Ф = 3 \times 23{,}3^2 \times 5 = 8143 \text{ Вт}.$$

Суммарная реактивная мощность

$$Q_\Sigma = 3Q_Ф = 3I_Ф^2 X_L = 3 \times 23{,}3^2 \times 8 = 13029 \text{ Вар}.$$

Суммарная полная мощность

$$S_\Sigma = \sqrt{3} \times 380 \times 23{,}3 = 15317 \text{ ВА}.$$

**Задача 2**

В трёхпроводную сеть трехфазного тока с линейным значением ЭДС, равным 380 В, включены три группы ламп с сопротивлениями $R_A$ = 20 Ом, $R_B$ =10 Ом, $R_C$ = 15 Ом (рисунок 3.2).

Определить: линейные токи $\dot{I}_A$, $\dot{I}_B$, $\dot{I}_C$, активную мощность трёхфазной системы.

$$\dot{E}_A = \frac{E_Л}{\sqrt{3}} = \frac{380}{\sqrt{3}} = 220 \text{ В};$$

$$\dot{E}_B = 220e^{-j120^\circ} \text{ В};$$

$$\dot{E}_C = 220e^{+j120^\circ} \text{ В}.$$

Рассчитаем проводимости отдельных фаз:

$$\dot{Y}_A = \frac{1}{R_A} = \frac{1}{20} = 0{,}05 \text{ См};$$

$$\dot{Y}_B = \frac{1}{R_B} = \frac{1}{10} = 0{,}1 \text{ См};$$

$$\dot{Y}_C = \frac{1}{R_C} = \frac{1}{15} = 0{,}067 \text{ См}.$$

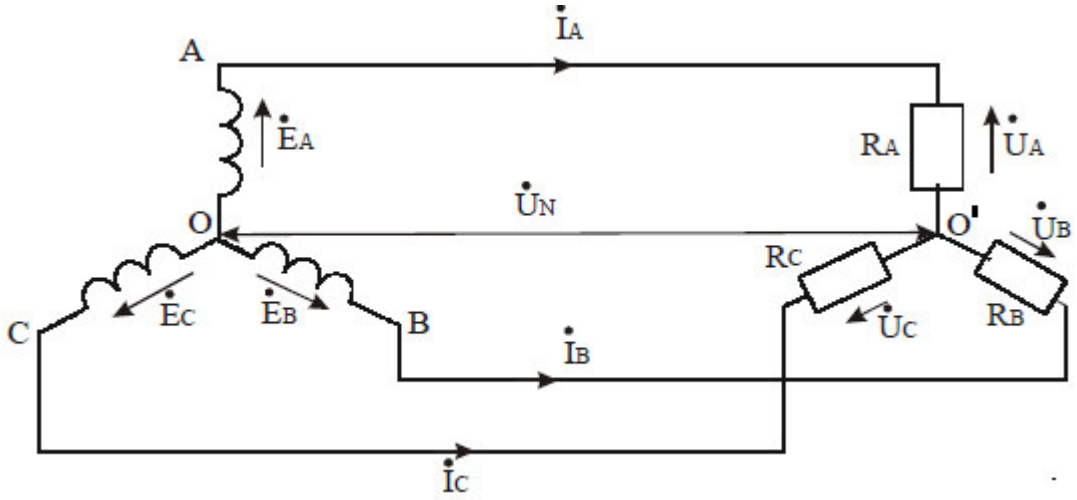



Рисунок 3.2. Схема замещения цепи

Напряжение между точками О и О′

$$\dot{U}_N = \frac{\dot{E}_A\dot{Y}_A + \dot{E}_B\dot{Y}_B + \dot{E}_C\dot{Y}_C}{\dot{Y}_A + \dot{Y}_B + \dot{Y}_C} = \frac{220\times 0{,}05 + 220e^{-j120^\circ}\times 0{,}1 + 220e^{+j120^\circ}\times 0{,}067}{0{,}05 + 0{,}1 + 0{,}067} =$$

$$= \frac{11 + 22e^{-j120^\circ} + 14{,}74e^{+j120^\circ}}{0{,}217} = \frac{11 + 22\cos(-120^\circ) + j22\sin(-120^\circ) + 14{,}74\cos 120^\circ + j14{,}74\sin 120^\circ}{0{,}217} =$$

$$= \frac{11 - 11 - j19 - 7{,}37 + j12{,}76}{0{,}217} = \frac{-7{,}37 - j6{,}24}{0{,}217} = (-33{,}96 - j28{,}75)\ \text{В}.$$

Фазные напряжения на зажимах нагрузки:

$$\dot{U}_A = \dot{E}_A - \dot{U}_N = 220 - (33{,}96 - j28{,}75) = (253{,}96 + j28{,}75)B;$$

$$\dot{U}_B = \dot{E}_B - \dot{U}_N = 220e^{-j120^\circ} = (= 33{,}96 - j28{,}75) = 220\cos(-120^\circ) +$$

$$+ j220\sin(-120^\circ) + 33{,}96 + j28{,}75 = -110 - j190 + 33{,}96 + j28{,}75 =$$

$$= (-76{,}04 - j161{,}25)\ \text{В};$$

$$\dot{U}_C = \dot{E}_C - \dot{U}_N = 220e^{j120^\circ} - (33{,}96 - j28{,}75) = 220\cos 120^\circ + j220\sin 120^\circ + 33{,}96 + j28{,}75 =$$

$$= -110 + j190{,}5 + 33{,}96 + j28{,}75 = (-76 + j219{,}25)\ \text{В}.$$

Определим токи в линиях:

$$\dot{I}_A = \dot{U}_A\dot{Y}_A = (253{,}96 + j28{,}75)\times 0{,}05 = 12{,}7 + j1{,}43 = 12{,}78e^{j6{,}4^\circ}\ \text{В};$$

$$\dot{I}_B = \dot{U}_B\dot{Y}_B = (-76{,}04 - j161{,}25)\times 0{,}1 = (-7{,}6 - j16{,}1) = 17{,}8e^{j244{,}7^\circ} = 17{,}8e^{-j115{,}3^\circ}\ \text{В};$$

$$\dot{I}_C = \dot{U}_C\dot{Y}_C = (-76 + j219{,}25)\times 0{,}067 = (-5{,}1 + j14{,}69) = 15{,}55e^{j109^\circ}\ \text{В}.$$

Суммарная активная мощность равна

$$P_\Sigma = I_A^2R_A^2 + I_B^2R_B^2 + I_C^2R_C^2 = 12{,}78^2\times 20 + 17{,}8^2\times 10 + 15{,}55^2\times 15 = 10062\ \text{Вт}.$$

**Задача 3**

К трехпроводной трехфазной сети подключен приемник, соединенный звездой, активная мощность которого P = 2900 Вт, напряжение Uп = 220 В и cosφ = 0,6.

Каждый провод линии, соединяющий генератор и приемник, имеет активное сопротивление Rл = 0,6 Ом и индуктивное сопротивление Xл = 1 Ом.

Обмотки генератора соединены звездой.

Найти напряжение на зажимах генератора, а также его активную и реактивную мощности. Определить падение напряжения и напряжения в линии.

***Решение***

Поскольку приемник симметричный, напряжение между нейтралями генератора и приемника равно нулю, поэтому каждую фазу цепи можно рассматривать независимо от других фаз и расчет проводить только для одной фазы, например А.

Фазный ток приемника равен линейному току:

$$I = \frac{P}{\sqrt{3}U_{П}\cos\varphi} = \frac{2900}{\sqrt{3}\times 220\times 0.6} = 12.7 \text{ А}.$$

Фазное напряжение приемника

$$U_{Ф} = \frac{U_{П}}{\sqrt{3}} = \frac{220}{\sqrt{3}} = 127 \text{ В}.$$

Сопротивления фаз приемника:

$$Z_{П} = \frac{U_{Ф}}{I} = \frac{127}{12,7} = 10 \text{ Ом};$$

$$R = Z_{П}\cos\varphi = 10\times 0,6 = 6 \text{ Ом};$$

$$X_{L} = Z_{П}\sin\varphi = 10\times 0,8 = 8 \text{ Ом}.$$

Сопротивления фазы (с учетом сопротивления линии)

$$Z = \sqrt{(R_{Л}+R)^2 + (X_{Л}+X_{L})^2} = \sqrt{(0,6+6)^2 + (1+8)^2} = 11,15 \text{ Ом}.$$

Фазное и линейное напряжения генератора

$$U_{A} = Z\times I = 11.15\times 12,7 - 141,5B;$$

$$U = \sqrt{3}U_{A} = 245B.$$

Падение напряжения в проводах линии

$$U_{ПР} = Z_{Л}\times I = \sqrt{R_{Л}^2 + X_{Л}^2}\times I = \sqrt{0,6^2 + 1^2}\times 12,7 = 14,8 \text{ В}.$$

Коэффициент мощности генератора

$$\cos\varphi = \frac{R_{Л}+R}{Z} = \frac{0,6+6}{11,15} = 0,592.$$

Активная мощность генератора

$$P = \sqrt{3}UI\cos\varphi = 3\times(R_{Л}+R)\times I^2 = 3,17 \text{ кВт}.$$

Полная мощность генератора

$$S = \sqrt{3}UI = \sqrt{3}\times 245\times 12,7 = 5390 \text{ ВА}.$$

Реактивная мощность генератора

$$Q = \sqrt{S^2 - P^2} = \sqrt{5,39^2 - 3,17^2} = 4,34 \text{ кВар}.$$

**V ЗАКЛЮЧИТЕЛЬНАЯ ЧАСТЬ**

В заключительной части коллективного алгоритмического тренинга преподаватель делает выводы, где отмечает положительные и отрицательные моменты в проведении практического занятия, а также дает краткие указания, советы по подготовке к следующему занятию.

В конце оставляется время для ответов на вопросы, возникшие у обучающихся.

**МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ**

по учебной дисциплине «Электротехника, электроника и схемотехника»
по разделу «Электрические цепи при постоянных и синусоидальных токах
и напряжениях»

**КОЛЛЕКТИВНЫЙ АЛГОРИТМИЧЕСКИЙ ТРЕНИНГ**
**(РЕШЕНИЕ ЗАДАЧ)**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки В.Г. Буцкая

**4184.01.01;МУ.02;2**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ**
**ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»**

**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**МОСКВА 2018**

Разработано Е.В. Корнеевой
Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для педагогических работников и обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

# ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»

## НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

Методические указания (МУ) подготовлены для педагогических работников и обучающихся, разработаны в соответствии с учебным планом по направлению подготовки 09.03.01 «Информатика и вычислительная техника» на основании требований Федерального государственного образовательного стандарта. МУ предназначены для изучения основ и методов программирования в рамках дисциплины «Программирование».

**О Г Л А В Л Е Н И Е**

Стр.

## ВВЕДЕНИЕ

**Цель** лабораторного практикума заключается в изучении основ и методов программирования, закреплении теоретических знаний и навыков, полученных на лекциях и практических занятиях.

***Особенность*** данного вида занятий заключается в последовательности осуществления практических и познавательных действий.

Каждое занятие содержит теоретические сведения, практические задания, методику их выполнения.

***Каждое занятие подразделяется на следующие части.***

***Первая – вступительная.*** Обучаемые знакомятся с темой и целью занятия, перечнем прикладного программного обеспечения для проведения исследования или моделирования.

***Вторая – теоретическая.*** Обучаемые самостоятельно изучают теоретические сведения по теме занятия.

***Третья – практическая.*** Обучаемые самостоятельно выполняют практические задания по теме занятия в соответствии с методикой их выполнения.

***Четвёртая – заключительная.*** Предназначена для подведения итогов, контроля качества усвоения материала. Подводятся итоги занятия, обучаемым выставляются оценки.

## УЧЕБНО-МЕТОДИЧЕСКОЕ, МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ

### а) Литература

**Основная учебная**


1. **Синицына, Т.Г.** Введение в программирование на языке С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.
2. **Синицына, Т.Г.** Основы объектно-ориентированного программирования в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.
3. **Синицына, Т.Г.** Программирование в С++. [Электронный ресурс]: рабочий учебник/ Синицына, Т.Г. - 2012. - http://lib.muh.ru.
4. **Львович И.Я.** Основы информатики [Электронный ресурс]: учебное пособие/ Львович И.Я., Преображенский Ю.П., Ермолова В.В.— Электрон. текстовые данные.— Воронеж: Воронежский институт высоких технологий, 2014.— 339 c. http://www.iprbookshop.ru/23359.— ЭБС «IPRbooks»
5. **Фарафонов А.С.** Программирование на языке высокого уровня [Электронный ресурс]: методические указания к проведению лабораторных работ по курсу «Программирование»/ Фарафонов А.С.— Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013.— 32 c.— http://www.iprbookshop.ru/22912.— ЭБС «IPRbooks»


**Дополнительная**


1. **Глазырина, И.Б.** Введение в программирование. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.
2. **Глазырина, И.Б.** Основные типы данных в Турбо Паскале. [Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.
3. **Глазырина, И.Б.** Модульное программирование.[Электронный ресурс]: рабочий учебник/ Глазырина, И.Б. - 2011. - http://lib.muh.ru.
4. **Глазырина, И.Б.** Динамические структуры. [Электронный ресурс]: рабочий учебник/ Лабзина, Т.А. - 2011. - http://lib.muh.ru.


### б) Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

– Программное обеспечение, являющееся частью электронной информационно-образовательной среды и базирующееся на телекоммуникационных технологиях:

- компьютерные обучающие программы.
- тренинговые и тестирующие программы.
- интеллектуальные роботизированные системы оценки качества выполненных работ.

– Информационные и роботизированные системы, программные комплексы, программное обеспечение для доступа к компьютерным обучающим, тренинговым и тестирующим программам:

- ПО «Комбат»;
- ПО «ЛиК»;
- ПК «КОП»;
- ИР «Каскад».

**в) Материально-техническое обеспечение:**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторный практикум № 1. Программирование алгоритмов линейной и разветвляющейся структуры. Составной оператор**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 1.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение правил записи оператора присваивания и процедур ввода/вывода в языке Паскаль, изучение процесса построения алгоритмов разветвляющейся структуры, программирование разветвляющихся алгоритмов с помощью оператора If…Then…Else и оператора Case…Of, изучение составного оператора Begin…End.

***Вводная часть***

*Описание интегрированной среды PascalABC*

Главное окно интегрированной среды PascalABC имеет следующий вид (рисунок 1).

Программа записывается в окне редактора, которое открывается при запуске или с помощью команды Файл – Новый (можно воспользоваться функциональными клавишами Ctrl-N).

Под окном редактора расположено окно вывода. Оно предназначено для вывода данных процедурами `Write` и `WriteLn`, а также для вывода сообщений об ошибках и предупреждений во время работы программы. Окно вывода может быть скрыто. Клавиша F5 показывает/скрывает окно вывода. Для скрытия окна вывода используется также клавиша Esc. Окно вывода обязательно открывается при любом выводе в него. Для очистки окна вывода следует нажать комбинацию клавиш Ctrl-Del.

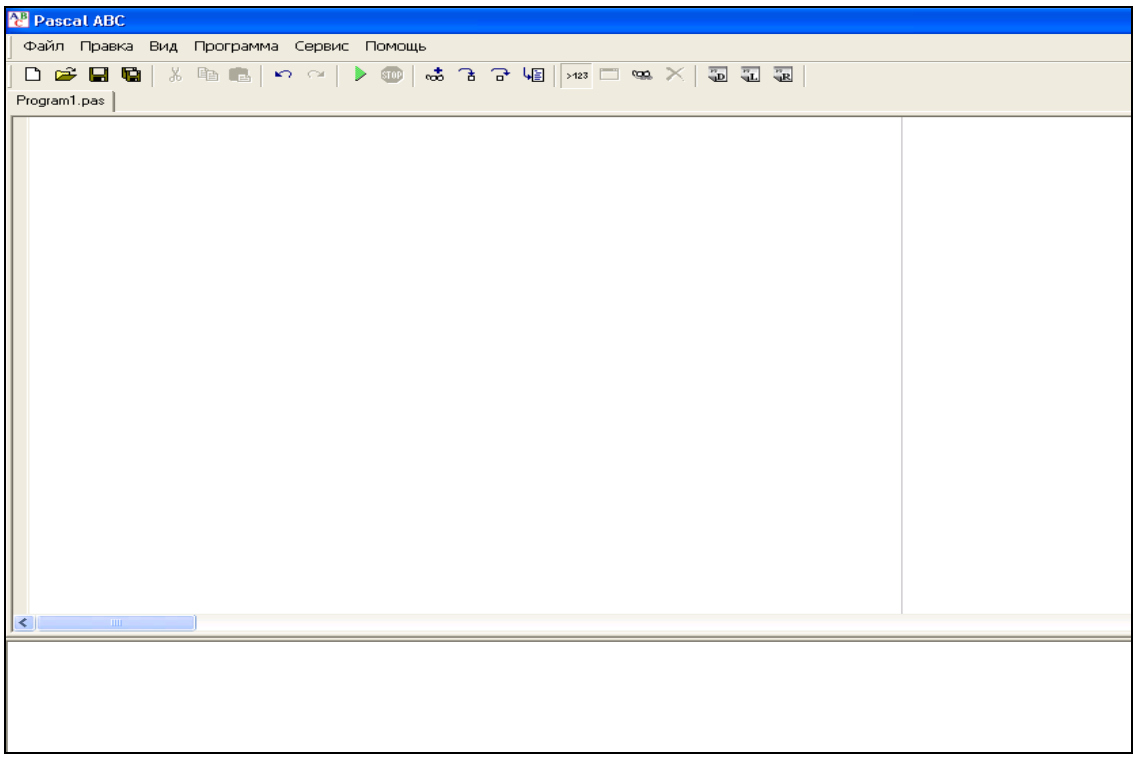



Рисунок 1. Главное окно интегрированной среды PascalABC

Окно ввода (рисунок 2) открывается при выполнении процедур `Read` и `ReadLn` в ходе работы программы.

Ввод данных в окно ввода сопровождается эхо-выводом в окно вывода. После нажатия клавиши Enter данные из окна ввода попадают в соответствующие переменные, окно ввода закрывается, и программа продолжает работать дальше.

Для запуска программы в текущем окне редактора следует нажать клавишу F9 или кнопку  панели инструментов.

Программа вначале компилируется во внутреннее представление, после чего, если не найдены ошибки, программа начинает выполняться. При выполнении программы кнопка запуска программы становится неактивной, кнопка останова программы, наоборот, активной, и в строке статуса отображается информация "Программа выполняется".

Выполнение программы можно в любой момент прервать нажатием комбинации клавиш Ctrl-F2 или кнопки  . При этом в окне вывода появится сообщение "Программа прервана пользователем".

*Структура программы на языке Паскаль*

Паскаль-программа является текстовым файлом с собственным именем и расширением .pas. Программа на Паскале состоит из двух частей: описательной части и операторной части. Программа может начинаться с заголовка, состоящего из служебного слова Program и имени, имеющего синтаксис идентификатора. Операторная часть начинается служебным словом Begin, а заканчивается программа служебным словом End. После этого оператора ставится точка.

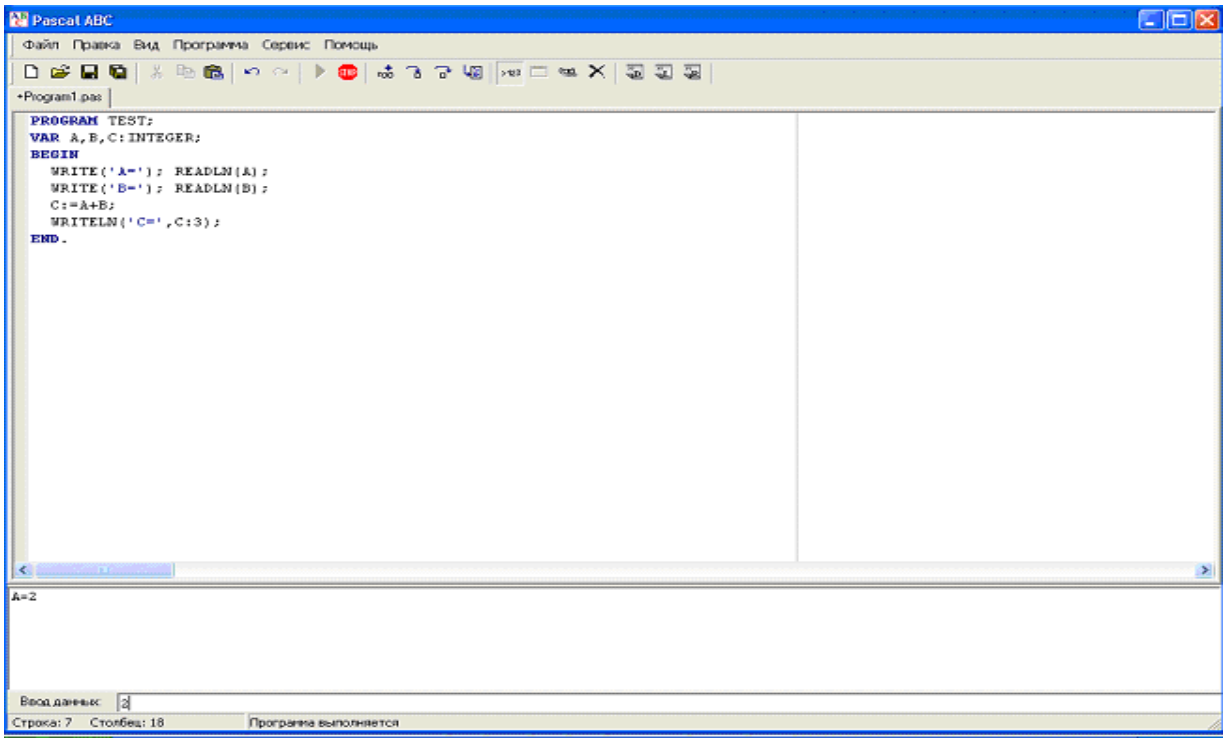



Рисунок 2. Работа с окном ввода

Синтаксис:

Program <Имя программы>;

<Описательная часть>

✍♏♑♓■

<Операторная часть>

End.

Операторы отделяются друг от друга точкой с запятой и располагаются по строкам произвольно. Опыт программирования показал, что программа лучше читается, если операторы располагаются по одному в строке.

Хороший стиль программирования предполагает использование комментариев в программе. Комментарии записываются в фигурных скобках и могут содержать любые символы. Вместо фигурных скобок могут использоваться символы (* и *). Например:

{Это комментарий}

(*Это тоже комментарий*)

*Оператор присваивания*

Основным оператором процедурного языка программирования является оператор присваивания. Синтаксис оператора:

<Переменная>:=<Выражение>;

Переменная и выражение должны иметь один и тот же тип. Двоеточие и знак равенства записываются слитно и называются знаком присваивания.

***Пример***

S := (a + b+ c)/3;

L := (S>0) OR (P < 100).

В этом примере S и соответствующее выражение в правой части имеют вещественный тип, а L и соответствующее выражение в правой части – логический тип.

Выражение записывается в строку, поэтому программист должен следить за последовательностью вычислений. Приоритет арифметических действий такой же, как в алгебре. Для изменения последовательности действий необходимо использовать скобки.

Пусть необходимо записать оператор присваивания для выражения

$$f = \frac{x^2 + 2y - \cos^2 x}{y^2 + \sqrt{2x^2 + 1}}.$$

Оператор присваивания:

F := (x*x + 2 * y – SQR(COS(x)))/(y * y + SQRT(2 * x * x + 1)).

Еще один пример:

$$k = \frac{e^{x+y}}{2x + 3y} \sin^2 x^3.$$

Оператор присваивания:

k := EXP(x+y)/(2*x+3*y)*SQR(SIN(x*x*x)).

Со стандартными функциями Паскаля можно познакомиться, вызвав справочную систему интегрированной среды.

*Процедуры ввода/вывода*

Вводить данные в Паскале можно при помощи двух стандартных процедур – Read и ReadLn. Процедуры используются следующим образом: сначала указывается имя процедуры, а затем в круглых скобках указывается список переменных, значения которых необходимо ввести. Переменные отделяются друг от друга запятыми.

***Пример***

Read(x, y, z);

ReadLn(a, b, c).

При выполнении этих процедур вводимые значения отделяются друг от друга одним или несколькими пробелами или переходом на следующую строку. В отличие от Read выполнение процедуры ReadLn завершается только после нажатия на клавишу Enter.

Рассмотрим эту разницу на примере. При выполнении следующего фрагмента программы

Read(a, b);

Read(c);

и при вводе данных

4 5 1

переменной *a* присвоится значение 4, переменной *b* – 5, а переменной *c* – 1. В случае же фрагмента

ReadLn(a, b);

ReadLn(c);

при вводе данных таким же образом переменной *a* присвоится значение 4, переменной *b* – 5. После этого ожидается ввод значения *c* на новой строке. Значение 1 будет проигнорировано.

Вывод данных обеспечивается двумя стандартными процедурами Write и WriteLn. В скобках после имени процедуры могут быть записаны константы, выражения, переменные.

***Пример***

Write(‘Корень уравнения= ', X);

Writeln(‘Сумма =’,Sum,’ Среднее значение=’,Sum/n);

Числа вещественных типов по умолчанию выводятся в экспоненциальной форме, что воспринимается обычным пользователем не лучшим образом. Для того чтобы вывести число в общепринятой форме, задается формат вывода

WriteLn(X:n:m);

где X – переменная вещественного типа, а *n* и *m* – выражения целого типа, характеризующие ширину поля вывода. Выражение *n* означает, что все число будет выравниваться по правому краю поля из *n* символов (недостающие символы слева заменяются пробелами), выражение *m* означает, что число будет выводиться с *m* знаками после запятой. При этом будет происходить округление абсолютной величины числа.

Для вывода целых чисел можно использовать формат

WriteLn(a:n);

В этом случае выводимое значение будет выравниваться по правому краю поля шириной в n символов.

Отличие WriteLn от Write состоит в том, что после вывода всех значений, перечисленных в скобках, производится переход на следующую строку.

*Правила построения алгоритма разветвляющейся структуры и записи программы с использованием оператора If…Then…Else*

Алгоритм решения задачи, как правило, представляет собой совокупность стандартных алгоритмических структур. Одной из таких структур является развилка (полная и неполная, рисунок 3). Разветвление применяется, когда в зависимости от условия нужно выполнить либо одно, либо другое действие.

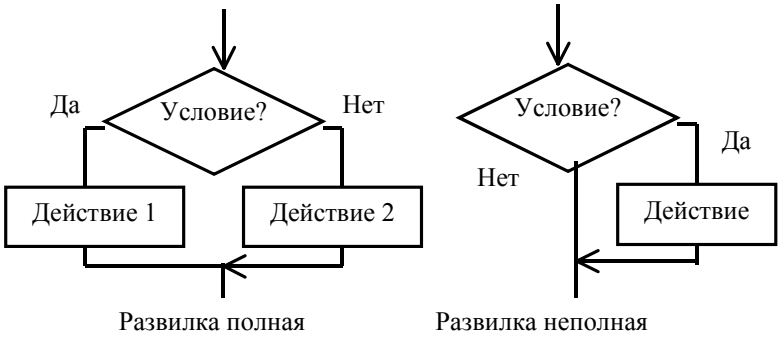



Рисунок 3. Схема разветвляющегося алгоритма

Для программирования проверки условия и выбора действия в зависимости от условия используются условные операторы.

Условный оператор:

If <логическое выражение> Then <Действие 1> Else <Действие 2>;

Если логическое выражение имеет значение True, то выполняется <Действие 1>. Если логическое выражение имеет значение False, то выполняется <Действие 2>.

Под обозначением <Действие 1> и <Действие 2> понимается один любой оператор языка. Если в зависимости от условия потребуется выполнить несколько операторов, то такие операторы нужно заключить в операторные скобки Begin и End. В Паскале любая последовательность операторов, находящаяся между словами Begin и End, считается одним оператором, называемым составным оператором.

Условный оператор может использоваться без части Else. В этом случае реализуется структура «развилка неполная».

If <логическое выражение> Then <Действие>;

Если логическое выражение имеет значение True, то выполняется оператор, стоящий за служебным словом Then, иначе осуществляется переход к оператору, следующему за условным оператором.

Пример записи оператора If:

```
If a<c Then m:=c Else m:=a;
If x<y Then min:=x;
If d<0 Then
Write('Корней нет')
   Else
   Begin
            x1:=(-b+sqrt(d))/(2*a);
            x2:=(-b-sqrt(d))/(2*a);
   End;
```

*Правила использования оператора Case…Of*

Если в алгоритме разветвляющейся структуры предполагается более двух вариантов (ветвей) расчета, а выбор варианта зависит от значения какой–либо одной переменной, то целесообразно использовать структуру «множественный выбор» (рисунок 4). Эта структура объединяет в себе несколько структур типа «развилка» и улучшает наглядность схемы алгоритма.

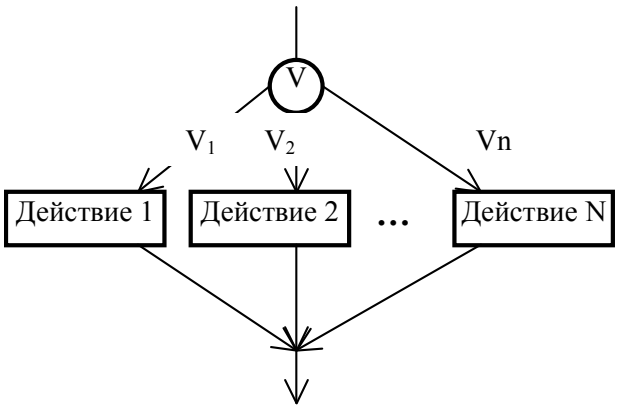



Рисунок 4. Структура множественного выбора

Решение задачи будет осуществляться по одной из ветвей алгоритма в зависимости от того, какое значение примет переменная V.

В программах такая структура реализуется с помощью оператора Case … Of.

Синтаксис оператора:

Case <Выражение порядкового типа> Of
<список констант 1> : <оператор 1>;
<список констант 2> : <оператор 2>;
…
<список констант N> : <оператор N>;
Else <Оператор>;
End;

Константы должны иметь такой же тип, что и выражение, следующее за служебным словом Case. Константы могут представлять собой интервал или разделяться запятыми.

Пример записи оператора Case:

```
Case Ch Of
'+' : Z := X + Y;
'-' : Z := X - Y;
'*' : Z := X * Y;
'/' : Z := X / Y;
Else
Stop := True;
End;
```

Переменная Ch имеет символьный тип. «Список выбора» организуют символьные константы '+', '-', '*', '/'. Часть Else может быть опущена.

***Практическая часть***

***Пример 1***

*Условие задачи*

Вычислить длину окружности и площадь круга по заданному радиусу.

Схема алгоритма представлена на рисунке 5.

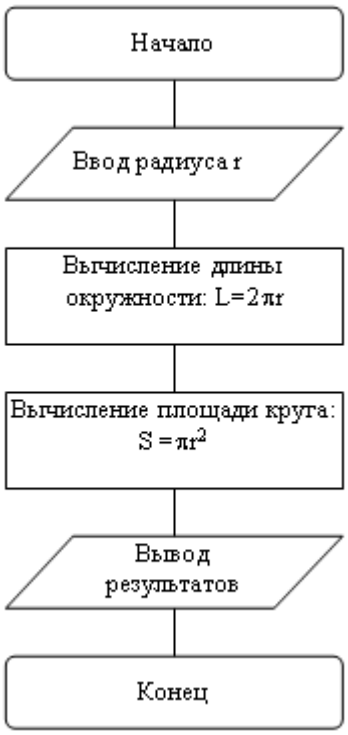



Рисунок 5. Схема алгоритма, вычисляющего длину окружности и площадь круга по заданному радиусу

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Circle;                          {начало программы}
Const                                    {описание констант}
  pi=3.14159;
Var                                      {описание переменных}
  r,l,s:Real;                            {r-радиус окружности, l-длина окружности}
                                         {s-площадь круга}
Begin                                    {начало операторной части}
  Write('Введите радиус: '); ReadLn(r);  {вывод подсказки ввода, ввод радиуса}
  l:=2*pi*r;                             {вычисление длины окружности}
  s:=pi*r*r;                             {вычисление площади круга}
  WriteLn('Длина окружности=',l:8:4);    {вывод результатов}
  WriteLn('Площадь круга=',s:8:4);
End.                                     {конец программы}
```

Результат работы программы представлен на рисунке 6.

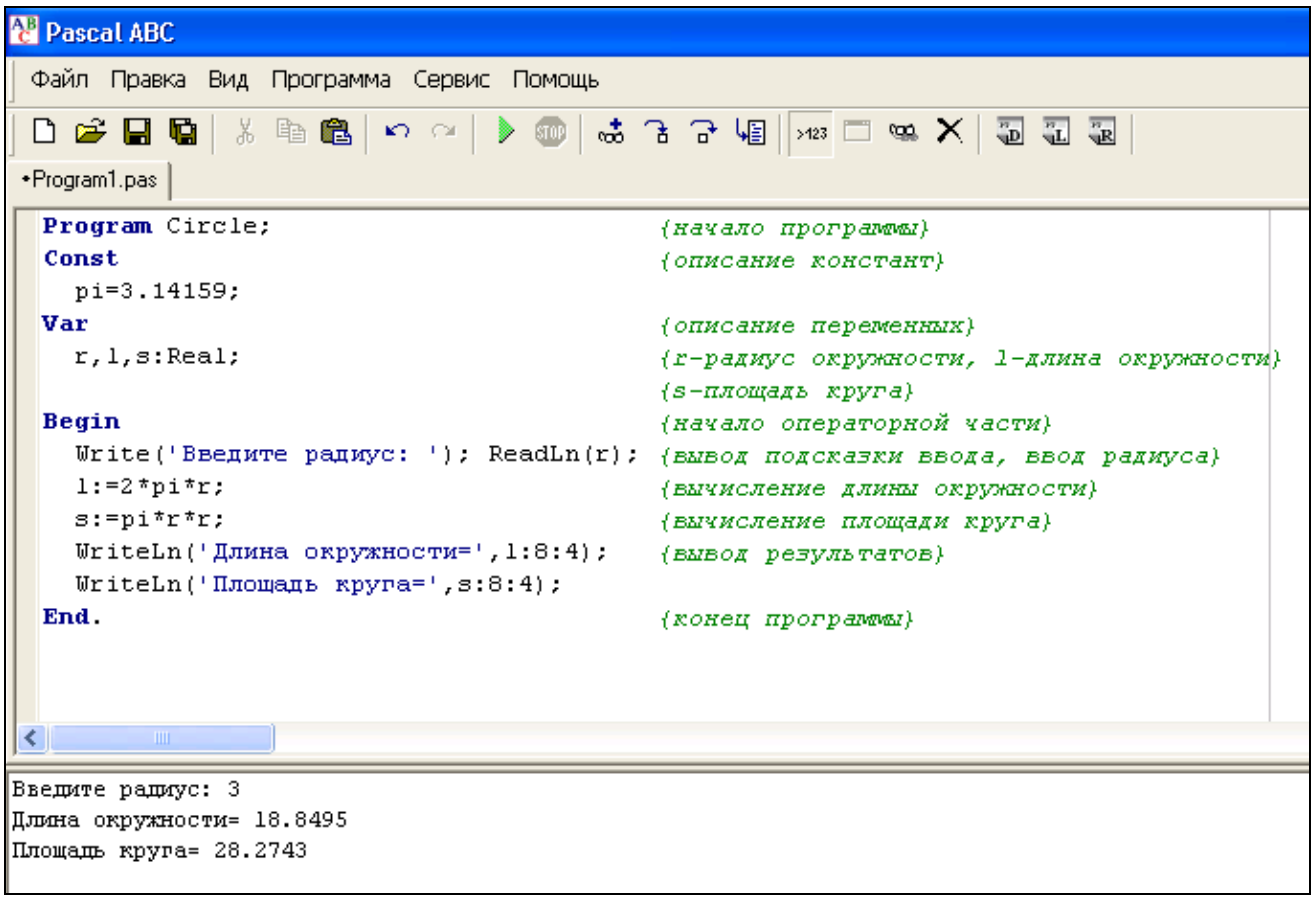



Рисунок 6. Результат работы программы, вычисляющей длину окружности
и площадь круга по заданному радиусу

***Пример 2***

*Условие задачи*

Какого роста цен за год можно ожидать, если правительство гарантирует, что инфляция в новом году составит p %?

Если за каждый месяц цены возрастут в 1 + p/100 раз, то за год рост цен составит $(1 + p/100)^{12}$ раз или прирост в процентах:

$$S = \left[\left(1 + \frac{p}{100}\right)^{12} - 1\right].100\ \%.$$

**Поясним формулу. За один месяц цены возрастут на p/100 и составят 1 + p/100. Для того чтобы сократить изложение, обозначим эту величину за X:**

$$X = 1 + p/100.$$

Единица – это условная цена единица товара. В следующем месяце цены возрастут на тот же процент, но уже от имеющихся цен на остаток месяца: X* p/100. Цены будут иметь значение X +X * p/100 или X*(1 + p/100).

$$X*(1 + p.100) = X^2.$$

В следующем (третьем месяце) цены будут иметь значение: $X^2 + X^2*p/100 = X^2*(1 + p/100) = X^3$.

И так далее. В конце 12-го месяца года цена единицы товара будет равна $X^{12}$ или$(1 + p/100)^{12}$. Очевидно, что рост цены будет равен $(1 + p/100)^{12} – 1$.

Чтобы получить прирост в процентах, необходимо умножить на 100 %.

Схема алгоритма представлена на рисунке 7.

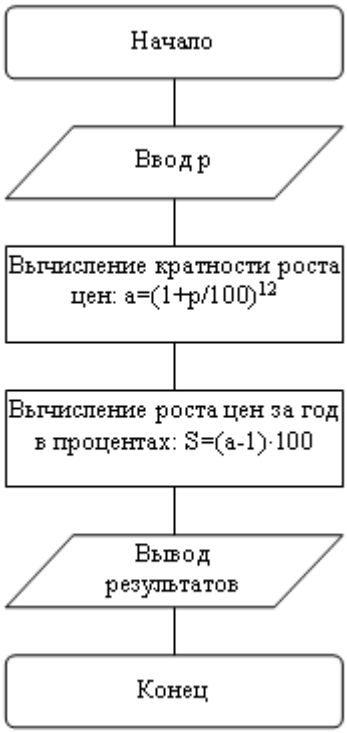



Рисунок 7. Схема алгоритма, вычисляющего предполагаемый рост цен при заданной инфляции

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Procent;                                    {Заголовок программы}
Var a,p,s: Real;                                    {Описание переменных}
{p - процент инфляции в месяц,
a - кратность роста цен, S - рост цен за год в процентах}
Begin                                               {Начало операторной части}
Write('Введите процент месячной инфляции: '); ReadLn(p);
                                                    {Вывод подсказки ввода. Ввод переменной p}
a := Power(1+p/100,12);
S := (a - 1) * 100;
WriteLn('Годовой прирост цен составит  ', a:10:2, ' раз или ', S:10:2, ' процентов');
                                                    {Процедура вывода}
End.
```

Результат работы программы представлен на рисунке 8.

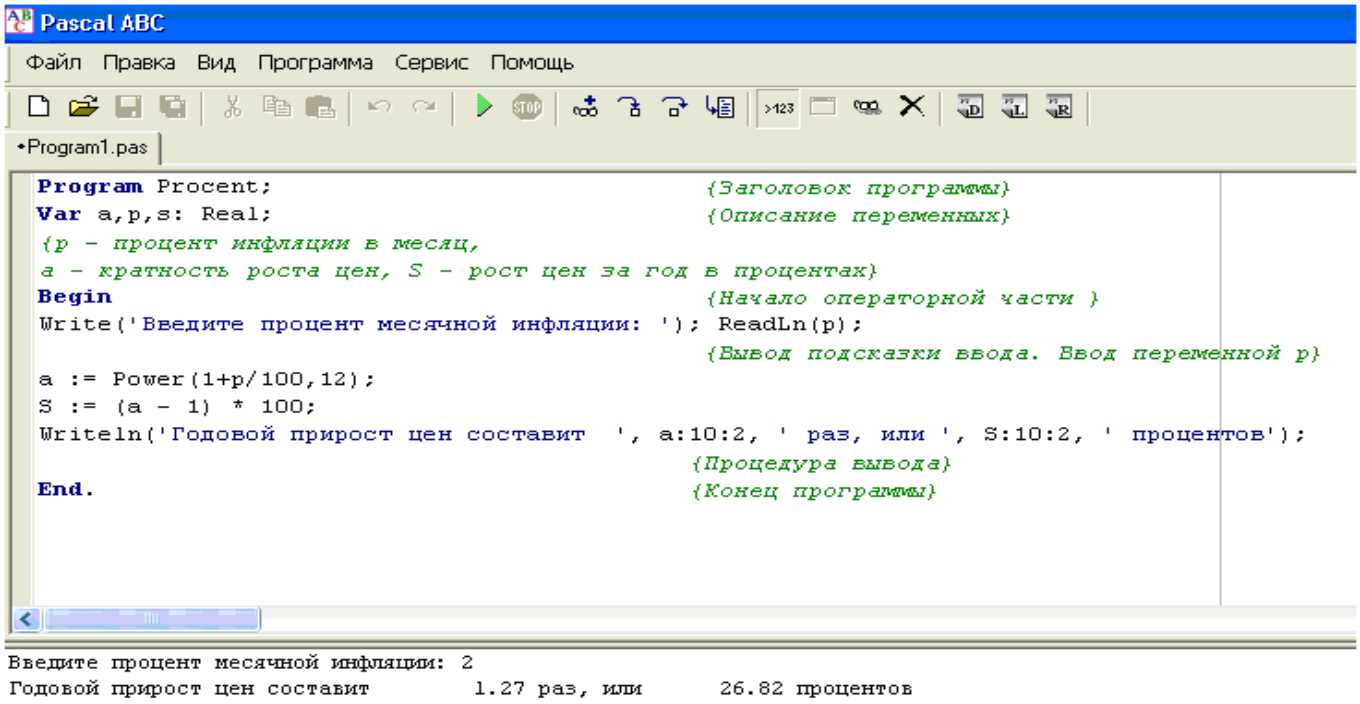



Рисунок 8. Результат работы программы, вычисляющей предполагаемый рост цен при заданной инфляции

***Пример 3***

*Условие задачи*

Даны числа X, Y, Z. Определить, что больше: сумма этих чисел или их произведение.

Схема алгоритма представлена на рисунке 9.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Comparat;
Var x,y,z,S,P:Real;
Begin
Write('Введите X: '); Readln(x);     {Ввод исходной информации}
Write('Введите Y: '); Readln(y);
Write('Введите Z: '); Readln(z);
S:=x+y+z;
P:=x*y*z;
If S > P Then WriteLn('Сумма больше, чем произведение X, Y, Z')
      {Оператор, исполняемый в случае,
      если логическое выражение имеет значение True}
Else WriteLn('Произведение больше или равно сумме X, Y, Z');
      {Оператор, исполняемый в случае,
      если логическое выражение имеет значение False}
End.
```

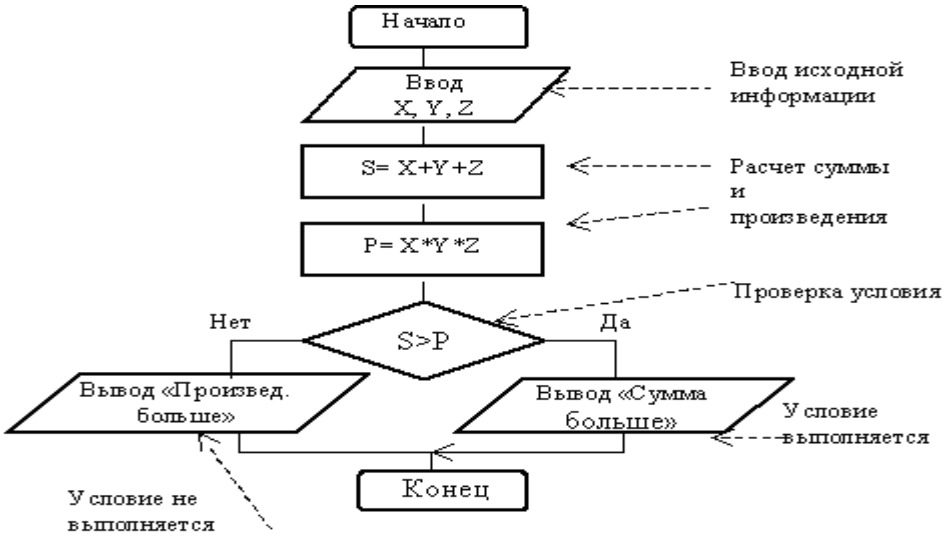



Рисунок 9. Схема алгоритма примера 3

Результат работы программы представлен на рисунке 10.

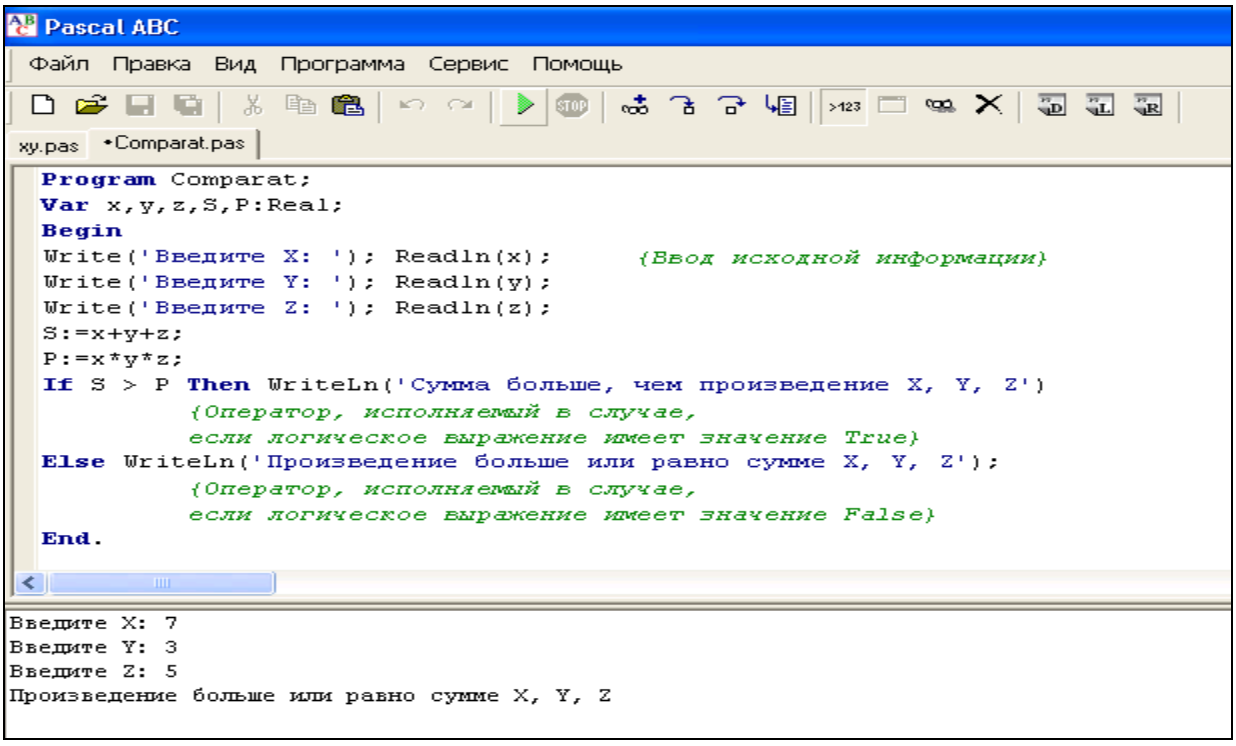



Рисунок 10. Результат работы программы, представленной в примере 3

***Пример 4***

*Условие задачи*

Даны два числа X и Y. Вычислить квадратные корни данных чисел, если оба значения больше нуля, и оставить числа без изменения, если это не так.

Схема алгоритма представлена на рисунке 11.

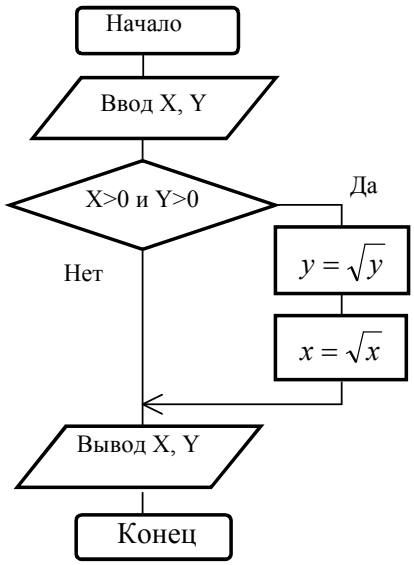



Рисунок 11. Схема алгоритма примера 4

Программа на Паскале в этом случае может иметь следующий вид:

```
Program XY;
Var x,y:Real;
Begin
Write('Введите x и y: '); Readln(x,y);
If (x>0) and (y>0) Then Begin                    {Если лог. выражение имеет значение True,
то выполняется составной оператор}
X := Sqrt(x);
Y := Sqrt(y);
End;                                                    {Условный оператор закончился}
Writeln('x=', x, ', y=', y);
End.
```

Результат работы программы представлен на рисунке 12.

***Пример 5***

*Условие задачи*

Написать программу для вывода дня недели.

Программа на Паскале может иметь следующий вид:

```
Program W;
Var number:Integer;
Begin
Write('Введите номер дня недели: '); ReadLn(number);
Case number of
1: WriteLn('понедельник');
```

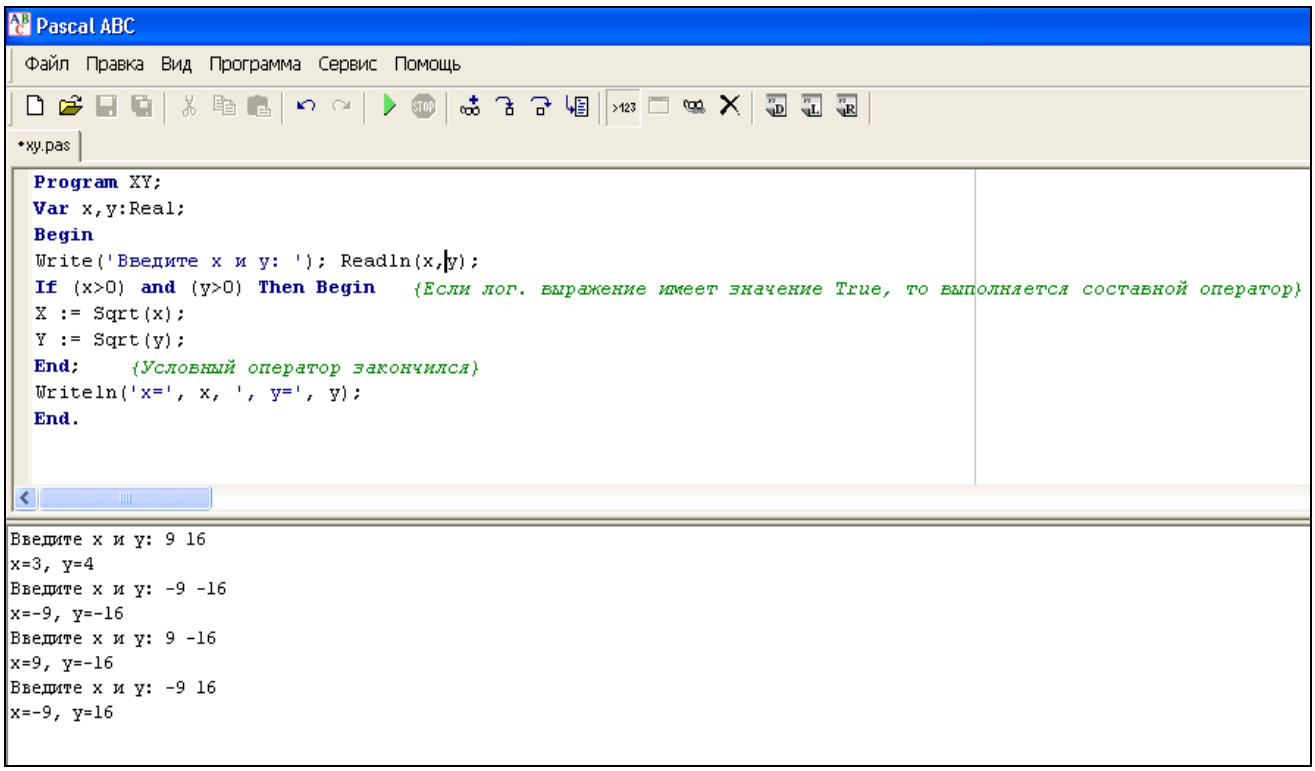



Рисунок 12. Результат работы программы, представленной в примере 4

2: WriteLn('вторник');
3: WriteLn('среда');
4: WriteLn('четверг');
5: WriteLn('пятница');
6: WriteLn('суббота');
7: WriteLn('воскресенье');
else WriteLn('недопустимое значение');
End;
End.

Результат работы программы представлен на рисунке 13.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу, определяющую функцию r:

$$r = \sqrt{S^2 + t};$$

$$S = \frac{5x^2 + 2e^y}{3x + \sin^2 y^3};$$

$$t = \frac{x^2 + 2y - 3}{2\sin^2 x + \frac{1}{x + y}}$$

при заданных значениях x и y. При выводе округлите результаты до двух знаков после запятой.

2. 1 верста = 500 саженей; 1 сажень = 3 аршина; 1 аршин = 16 вершков; 1 вершок = 44,45 мм. Длина некоторого отрезка составляет 5000 метров. Составить программу, которая переводит ее в русскую неметрическую систему – версты. При выводе округлите результат до двух знаков после запятой.

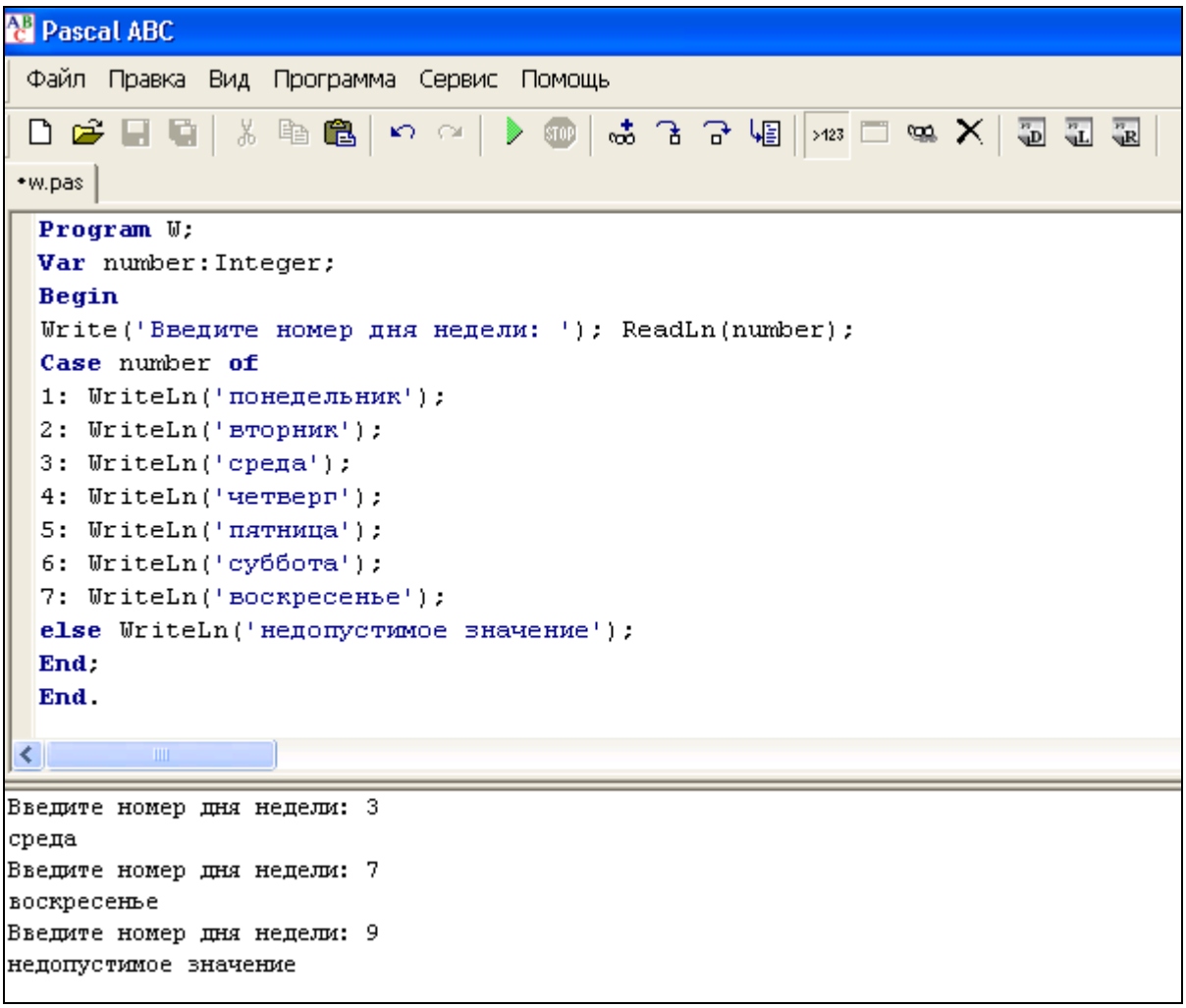



Рисунок 13. Результат работы программы, представленной в примере 5

3. Даны две точки $A(x_1,y_1)$ и $B(x_2,y_2)$. Составить программу, определяющую, которая из точек находится ближе к началу координат.

Примечание: $r = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ .

4. Дано натуральное число M. Если оно делится на 7, вывести на экран ответ M=7k (где k – соответствующее частное); если остаток от деления на 7 равен 1, вывести M=7k+1; если остаток от деления на 7 равен 2, вывести M=7k+2; если остаток от деления на 7 равен 3, вывести M=7k+3 и т.д. Например, 14=7*2 или 30=7*4+2. Составить программу с использованием оператора Case…Of.

*Вариант 2*

1. Составить программу, определяющую функцию u:

$$u = v^2 + w;$$

$$v = \frac{3x + \ln^2 2y}{e^{x/2} - \cos x^2};$$

$$w = \sqrt{x^2 + y^2} + \frac{2y + 3}{5tgx}$$

при заданных значениях x и y. При выводе округлите результаты до двух знаков после запятой.

2. В равнобедренном прямоугольном треугольнике известна высота h=3.94, опущенная на гипотенузу. Составить программу, определяющую стороны треугольника. При выводе округлите результаты до двух знаков после запятой.

Примечание: гипотенуза = 2h; $2a^2 = 4h^2$; $a = h\sqrt{2}$

3. Составить программу для вычисления функции $F(x) = \begin{cases} x^2 + 3x + 9, \text{если } x \le 3 \\ \dfrac{\sin x}{x^2 - 9}, \text{если } x > 3 \end{cases}$. При выводе округлите результат до двух знаков после запятой.

4. Дано натуральное число X. Если оно делится на 7, вывести на экран «воскресенье»; если остаток от деления на 7 равен 1, вывести «понедельник»; если остаток от деления на 7 равен 2, вывести «вторник»; если остаток от деления на 7 равен 3, вывести «среда»; если остаток от деления на 7 равен 4, вывести «четверг»; если остаток от деления на 7 равен 5, вывести «пятница»; если остаток от деления на 7 равен 6, вывести «суббота». Составить программу с использованием оператора Case…Of.

*Вариант 3*

1. Составить программу, определяющую функцию v:

$$v = \sqrt{w} - a;$$

$$w = \frac{2x + e^y}{\cos x + y};$$

$$a = \frac{1 + tgx^2}{x^3 - y}$$

при заданных значениях x и y. При выводе округлите результаты до двух знаков после запятой.

2. Заданы действительная и мнимая части комплексного числа Z = X + I Y; X=3.4; Y=4.6. Составить программу, преобразующую его в тригонометрическую форму в виде выражения Z = r(cos φ + i sin φ). При выводе округлите результаты до двух знаков после запятой.

Примечание: $r = \sqrt{x^2 + y^2}$; $\varphi = \text{arctg}\,\frac{y}{x}$.

3. Заданы три натуральных числа. Составить программу, определяющую, является ли среднее арифметическое этих чисел целым числом.

4. Дано натуральное число X. Составить программу, определяющую значение Y. Если X делится на 5, $Y=X^{-5}$; если остаток от деления X на 5 равен 1, $Y=X^{-4}$; если остаток от деления X на 5 равен 2, $Y=X^{-4}$; если остаток от деления X на 5 равен 3, $Y=X^{-4}$; если остаток от деления X на 5 равен 4, $Y=X^{-1}$. Написать программу с использованием оператора Case…Of. При выводе округлите результат до четырех знаков после запятой.

*Вариант 4*

1. Составить программу, определяющую функцию t:

$$t = \ln(u^2 + v^2);$$

$$u = x + 5\cos 2y;$$

$$v = \frac{e^x + 2y + 3}{3x + \sqrt{x^2 + y^2}}$$

при заданных значениях x и y. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу, вычисляющую расстояние между двумя точками (1,2) и (10,15). При выводе округлите результат до двух знаков после запятой.

Примечание: $r = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

3. Даны три числа. Составить программу, которая возводит в квадрат те из них, значения которых неотрицательны, и находит произведение полученных чисел.

4. Дано натуральное число Z. Составить программу, определяющую значение Y. Если Z делится на 6, Y=Z; если остаток от деления Z на 6 равен 1, $Y=Z^2$; если остаток от деления Z на 6 равен 2, $Y=Z^3$; если остаток от деления Z на 6 равен 3, $Y=Z^{-3}$; если остаток от деления Z на 6 равен 4, $Y=Z^{-2}$; если остаток от деления Z на 6 равен 5, $Y=Z^{-1}$. Написать программу с использованием оператора Case…Of.

*Вариант 5*

1. Составить программу, определяющую функцию h:

$$h = k^2 + l^2;$$

$$k = \frac{3e^{-x} + 2y}{x + \sqrt{x + y^2}};$$

$$l = \frac{x^2 + 2y - 3}{x \ln y + 1}$$

при заданных значениях x и y. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу, определяющую объем и площадь боковой поверхности цилиндра с заданным радиусом основания r = 4 и высотой h = 6. При выводе округлите результаты до двух знаков после запятой.

Примечание: $V = \pi r^2 h$; $S = 2\pi rh$; $\pi = 3.14$.

3. Составить программу, определяющую, пройдет ли график функции $y = ax^2 + bx + c$ через заданную точку с координатами (m, n).

4. Дано натуральное число N. Если оно делится на 4, вывести на экран ответ N = 4k (где k – соответствующее частное); если остаток от деления на 4 равен 1, вывести N = 4k + 1; если остаток от деления на

4 равен 2, вывести N = 4k + 2; если остаток от деления на 4 равен 3, вывести N = 4k + 3. Например, 12 = 4*3 или 22 = 4 * 5 + 2. Составить программу с использованием оператора Case…Of.

### Лабораторный практикум № 2. Программирование алгоритмов циклической структуры

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 1.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство с процессом построения алгоритмов циклической структуры, изучение правил записи операторов цикла в языке Паскаль, программирование циклических алгоритмов.

***Вводная часть***

*Организация циклических вычислений*

Цикл – типичная структура, характерная для программ, реализуемых на ЭВМ. Возможны три способа организации циклических структур алгоритмов (рисунок 14).

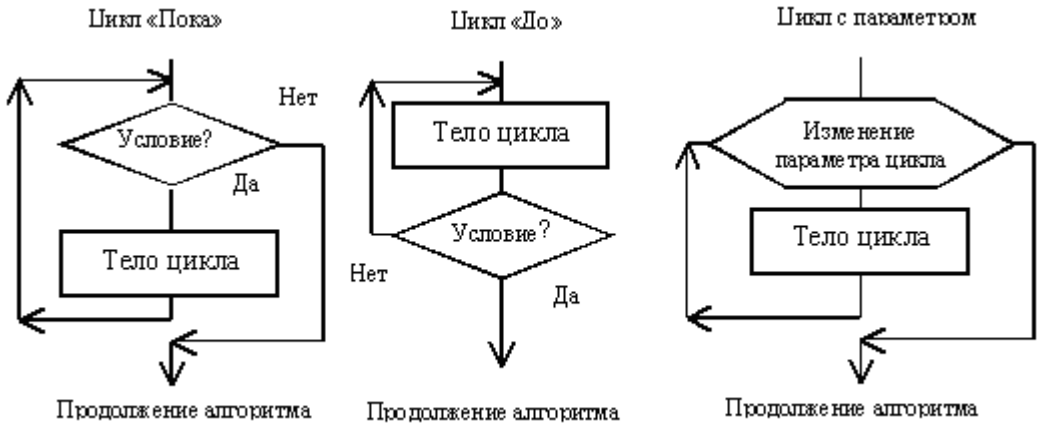



Рисунок 14. Способы организации циклических структур алгоритмов

Тело цикла – это повторяющаяся последовательность действий. Логический блок предназначен для управления циклом. Логический блок определяет количество проходов в цикле.

В цикле «Пока» условие окончания цикла расположено до тела цикла, поэтому возможны варианты, когда цикл не выполнится ни разу. В цикле «До» условие окончания цикла расположено после тела цикла. Это означает, что тело цикла выполниться хотя бы один раз. В Паскале эти структуры реализуются с помощью оператора цикла с предусловием и оператора цикла с постусловием.

В цикле с параметром число повторений заранее известно. Данный цикл предусматривает повторное выполнение некоторого оператора с одновременным изменением по правилу арифметической прогрессии значения управляющей переменной (параметра) цикла. В Паскале эта структура реализуется с помощью оператора цикла с параметром.

*Синтаксис операторов цикла в языке Паскаль*

Синтаксис оператора цикла с предусловием:

While <логическое выражение> Do <оператор>;

До тех пор, пока логическое выражение имеет значение True, выполняется тело цикла. Телом цикла может быть любой оператор языка, в том числе и составной.

Синтаксис оператора цикла с постусловием:

Repeat <последовательность операторов> Until <логическое выражение>;

Тело цикла расположено между служебными словами Repeat и Until. Это любая последовательность операторов языка. Операторы выполняются в цикле до тех пор, пока логическое выражение имеет значение False. Как только выражение примет значение True, цикл закончит свою работу и осуществится переход к выполнению следующего оператора, расположенного после оператора цикла.

Синтаксис оператора цикла с параметром:

For <идентификатор> := <выражение1> To <выражение2> Do <оператор>;

Идентификатор переменной и выражение должны иметь один и тот же порядковый тип. Переменная изменяется в цикле от значения <выражение1> до значения <выражение2> с шагом 1. Эта переменная управляет циклом и называется параметром цикла. В цикле выполняется один любой оператор языка, в том числе и составной. Шаг изменения параметра цикла постоянен и равен 1. Возможна другая интерпретация оператора:

For <идентификатор> := <выражение1> Downto <выражение2> Do <оператор>;

В этом случае шаг изменения параметра равен – 1.

Оператор For чаще используется в случаях, когда при организации циклов необходимо использовать счетчик.

***Практическая часть***

***Пример 1***

*Условие задачи*

Написать программу табулирования функции Y = x – sin x – 0.25 с помощью оператора цикла с предусловием.

Схема алгоритма представлена на рисунке 15.

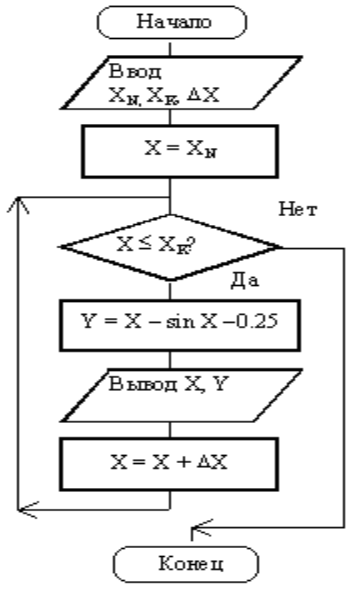



Рисунок 15. Схема алгоритма табулирования функции (пример 1)

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Tabl1;
Var x, xn, xk, dx, y: Real;
{x - текущее значение аргумента}
{xn - начальное значение аргумента}
{xk - конечное значение аргумента}
{dx - шаг изменения аргумента}
{y - значение функции}
```

```
Begin
Write('Начальное значение аргумента: '); Readln(xn);
Write('Конечное значение аргумента: '); Readln(xk);
Write('Шаг изменения аргумента: '); Readln(dx);
Writeln('======================');
Writeln('    X        Y');
x := xn;                          {Присвоили начальное значение аргументу}
While x<= xk Do                   {Цикл выполняется до тех пор, пока выполняется условие}
Begin                             {В цикле выполняется один составной оператор}
  y := x - sin (x) - 0.25;        {Рассчитали значение функции}
  Writeln ( x:8:2, y:10:2);       {Вывели результат на экран}
  x := x + dx;                    {Текущее значение аргумента увеличили на шаг}
End;                              {Конец тела цикла}
Writeln('======================');
End.
```

Результат работы программы представлен на рисунке 16.

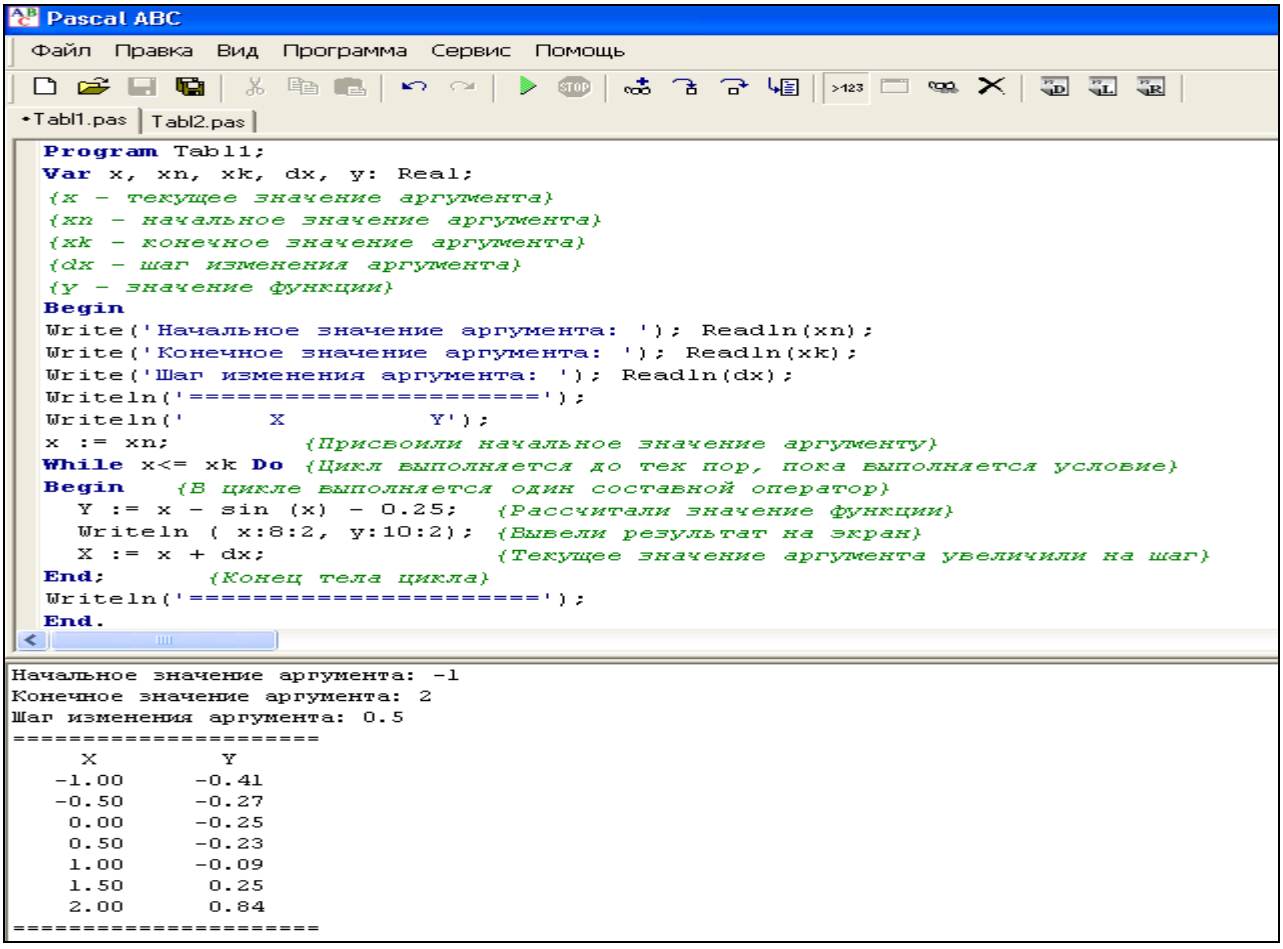



Рисунок 16. Результат работы программы табулирования функции (пример 1)

***Пример 2***

*Условие задачи*

Написать программу табулирования функции Y = x – sin x – 0.25 с помощью оператора цикла с постусловием.

Схема алгоритма представлена на рисунке 17.

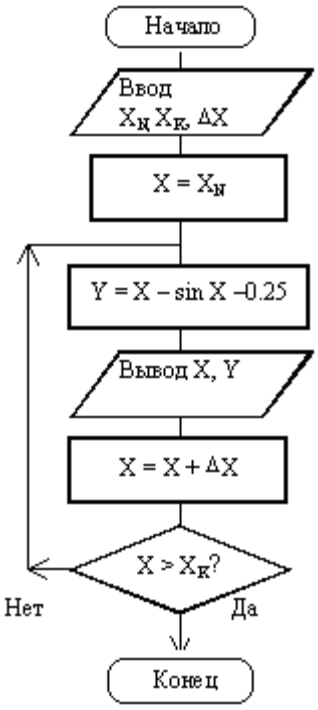



Рисунок 17. Схема алгоритма табулирования функции (пример 2)

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Tabl2;
Var x, xn, xk, dx, y: Real;
Begin
Write('Начальное значение аргумента '); Readln(xn);
Write('Конечное значение аргумента '); Readln(xk);
Write('Шаг изменения аргумента '); Readln(dx);
Writeln('======================');
Writeln('    X        Y');
x := xn;                                  {Присвоили начальное значение аргументу}
Repeat
{Начало тела цикла}
Y := x - sin (x) - 0.25;                  {Рассчитали значение функции}
Writeln ( x:8:2, y:10:2);                 {Вывели результат на экран}
X := x + dx;                              {Текущее значение аргумента увеличили на шаг}
                                  {Конец тела цикла}
Until x>xk;                               {Проверка условия окончания цикла}
Writeln('======================');
End.
```

Результат работы программы представлен на рисунке 18.

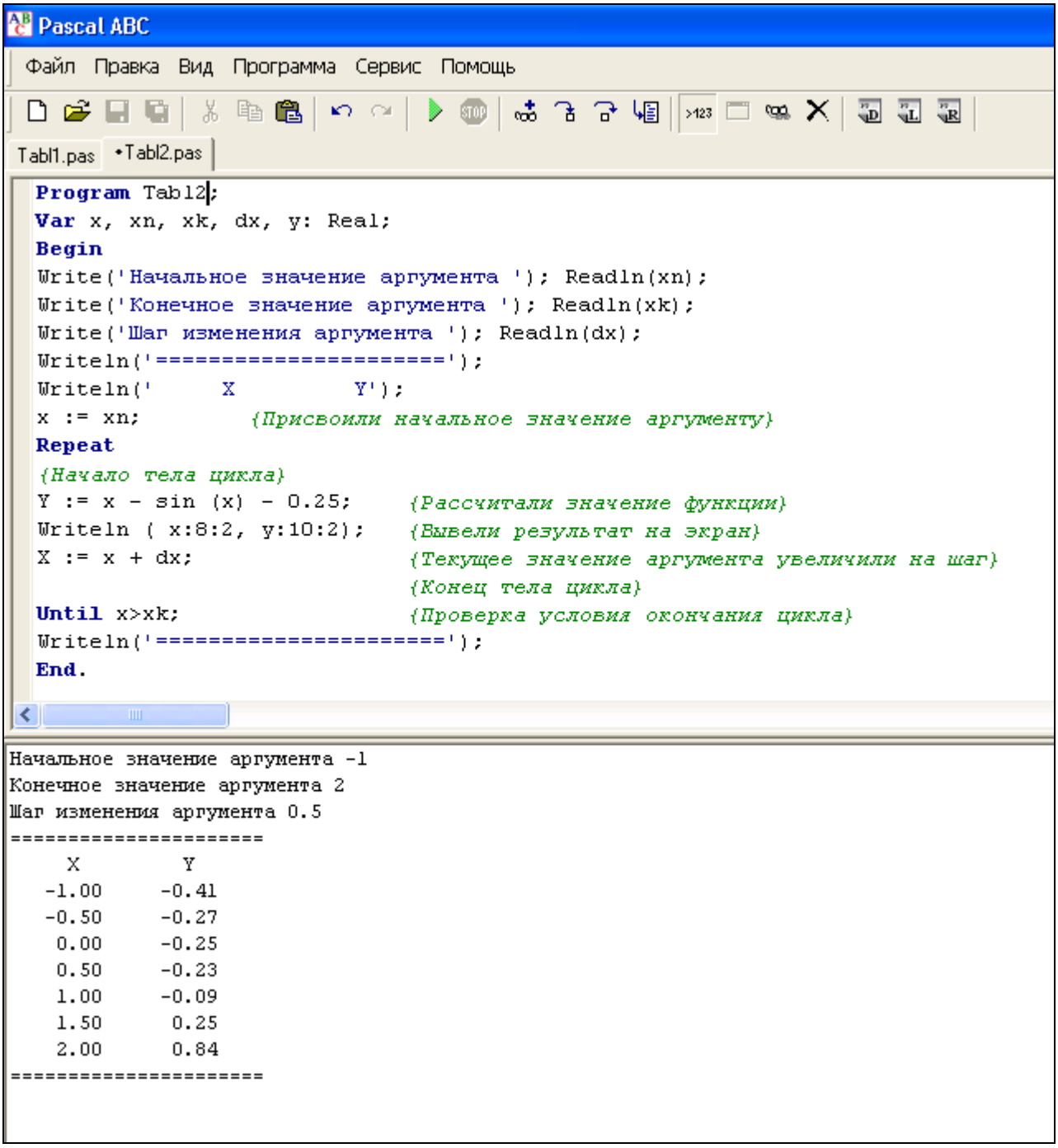



Рисунок 18. Результат работы программы табулирования функции (пример 2)

***Пример 3***

*Условие задачи*

Вычислить сумму квадратов первых N чисел натурального ряда.

Схема алгоритма представлена на рисунке 19.

Параметром цикла является переменная i. Значение переменной i изменяется от 1 до N с шагом 1. Каждое значение i, возведенное в квадрат, прибавляется к переменной Sum и результат присваивается значению Sum. Таким образом, в переменной Sum накапливается сумма квадратов i. Так как при первом проходе значение Sum в правой части выражения должно быть равно нулю, до начала цикла значение Sum обнуляется. Тот же прием используется при подсчете произведения. Очевидно, что начальное значение произведения должно равняться 1.

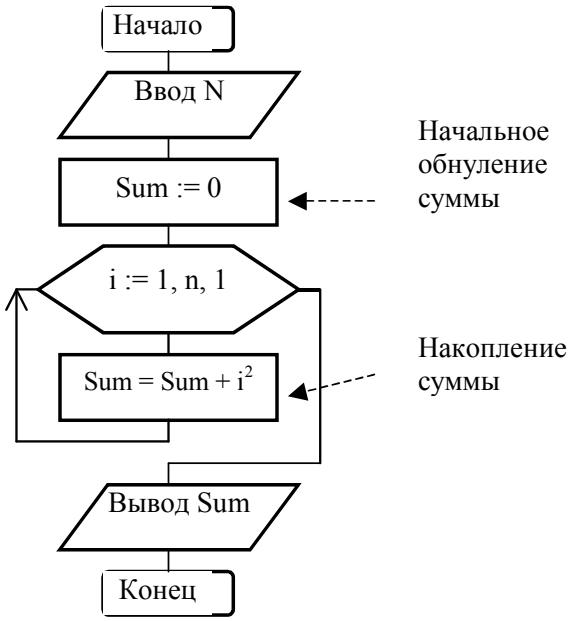



Рисунок 19. Схема алгоритма примера 3

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Summa;
Var i, n, Sum: integer;
Begin
Write('n= '); Readln(n);
Sum := 0;                    {Начальное обнуление суммы}
For i := 1 To n Do
  Sum := Sum + i*i;
 {Тело цикла - оператор присваивания для накопления суммы}
Writeln('Сумма= ',Sum);
End.
```

Результат работы программы представлен на рисунке 20.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу вычисления значений функции $F(x) = x - \sin x$ на отрезке $[0{,}4;0{,}9]$ с шагом 0,1. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу вычисления $S = \sum_{i=1}^{n} \frac{1}{i^2}$. При выводе округлите результат до двух знаков после запятой.

*Вариант 2*

1. Составить программу вычисления значений функции $F(x) = 2tg\frac{x}{2} + 1$ на отрезке $[-1;-0{,}4]$ с шагом 0,1. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу вычисления $\prod_{i=2}^{n} \frac{i+1}{i+2}$. При выводе округлите результат до двух знаков после запятой.

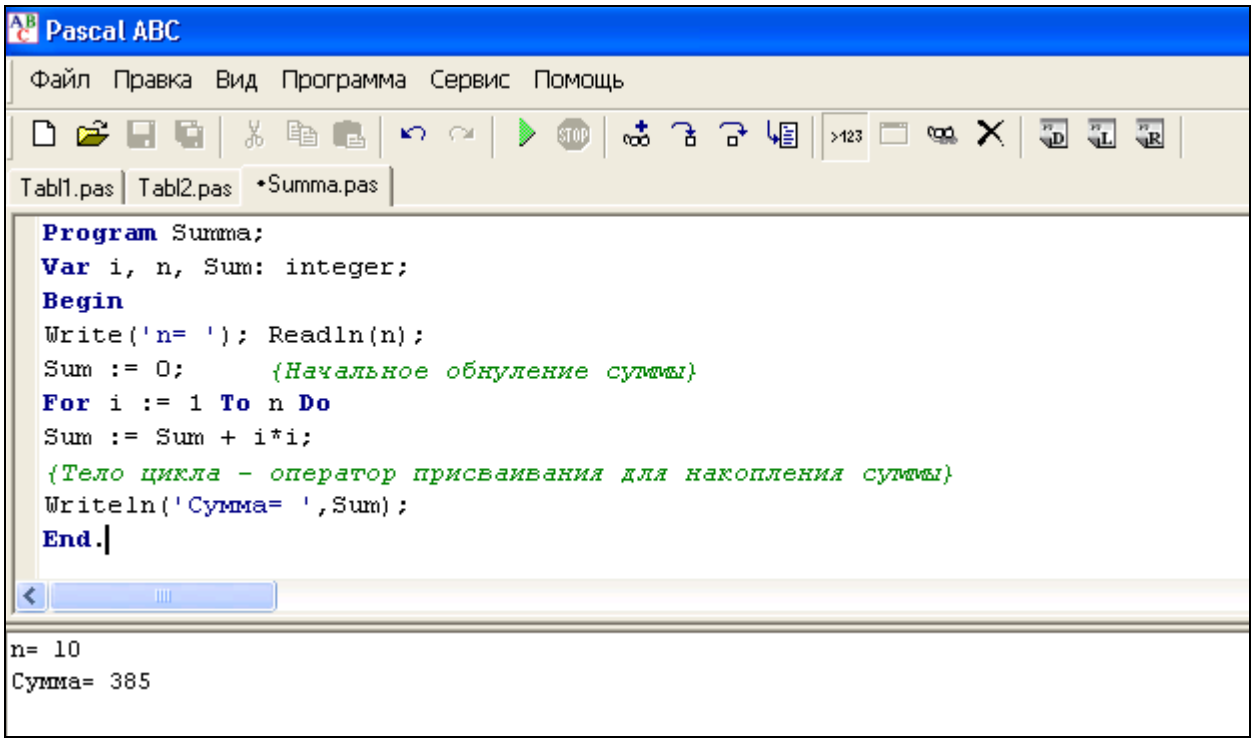



Рисунок 20. Результат работы программы, представленной в примере 3

*Вариант 3*

1. Составить программу вычисления значений функции $F(x) = 7\sin^2 x - \frac{1}{2}\cos x$ на отрезке $[10{,}1;10{,}7]$ с шагом 0,1. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу вычисления $S = \sum_{k=1}^{n} \frac{1}{(2k+1)^2}$. При выводе округлите результат до двух знаков после запятой.

*Вариант 4*

1. Составить программу вычисления значений функции $F(x) = \frac{1}{2}\sin\frac{x}{4} + 1$ на отрезке $[-18;-12]$ с шагом 1. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу вычисления $\prod_{k=2}^{n} (\frac{k}{k+1} - \cos k)$. При выводе округлите результат до двух знаков после запятой.

*Вариант 5*

1. Составить программу вычисления значений функции $F(x) = 2\cos\sqrt{x} + 0{,}5$ на отрезке $[21;28]$ с шагом 1. При выводе округлите результаты до двух знаков после запятой.

2. Составить программу вычисления $S = \sum_{i=1}^{n} \frac{i}{(2+i)^2}$. При выводе округлите результат до двух знаков после запятой.

**Лабораторный практикум № 3. Программирование задач накопления суммы и произведения элементов последовательности. Программирование итерационных алгоритмов**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 1.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство с процессом построения алгоритмов с неизвестным числом повторений, составления программ, использующих рекуррентные формулы.

***Вводная часть***

*Организация итерационных алгоритмов*

Циклические вычисления могут иметь либо заранее известное количество шагов, либо они выполняются до достижения некоторого условия. Циклические вычисления с неизвестным числом повторений, в которых число повторений определяется требуемой точностью вычислений, называются итерационными вычислениями. Повторение последовательности операторов с проверкой условия в начале каждого прохода цикла называется итерацией.

Рассмотрим пример. Необходимо рассчитать экспоненту с точностью ε путем разложения ее в ряд

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$$

Будем вычислять частичную сумму ряда правой части $S_n = \sum_{i=0}^{n} \frac{x^i}{i!}$ до тех пор, пока очередное слагаемое $\frac{x^i}{i!}$ не станет меньше погрешности ε.

Для вычисления каждого слагаемого ряда требуется возведение в степень и вычисление факториала (это дополнительный цикл). Часто в задачах для вычисления очередного слагаемого удобно рекуррентно использовать предыдущее слагаемое, а не организовывать дополнительный (внутренний) цикл. В данной задаче каждое очередное слагаемое можно рекуррентно вычислить через предыдущее: $U_i = \frac{X}{i} U_{i-1}$, что требует всего двух операций. Такая форма записи, в которой каждый следующий элемент расчета может быть получен из предыдущего, называется рекуррентной формой.

При построении таких алгоритмов полезно подстраховаться от зацикливания («вечного цикла»). Зацикливания могут возникнуть из–за ошибок в программе или вследствие накопления погрешностей. Чтобы избежать зацикливания, достаточно поставить лимит числа повторений цикла.

*Оператор безусловного перехода Goto*

Принудительный выход из цикла в программе на Паскале можно организовать, используя оператор безусловного перехода Goto. Синтаксис оператора:

Goto <Метка>;

После выполнения такого оператора управление программой передается оператору, помеченному меткой. Меткой может быть идентификатор или целое число без знака до 9999. Метка должна быть описана в разделе описания меток программы, который начинается ключевым словом Label. В тексте программы метка ставится перед оператором, который должен быть выполнен, и сразу после метки ставится двоеточие.

***Практическая часть***

*Пример 1*

*Условие задачи*

Рассчитать экспоненту с точностью ε путем разложения ее в ряд: $e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + ... + \frac{x^n}{n!} + \ldots$

При построении алгоритма примем следующие обозначения: S – искомая сумма, U – текущее слагаемое, получаемое рекуррентно, n – определяет число шагов и используется для расчета факториала в знаменателе слагаемого.

Схема алгоритма представлена на рисунке 21.

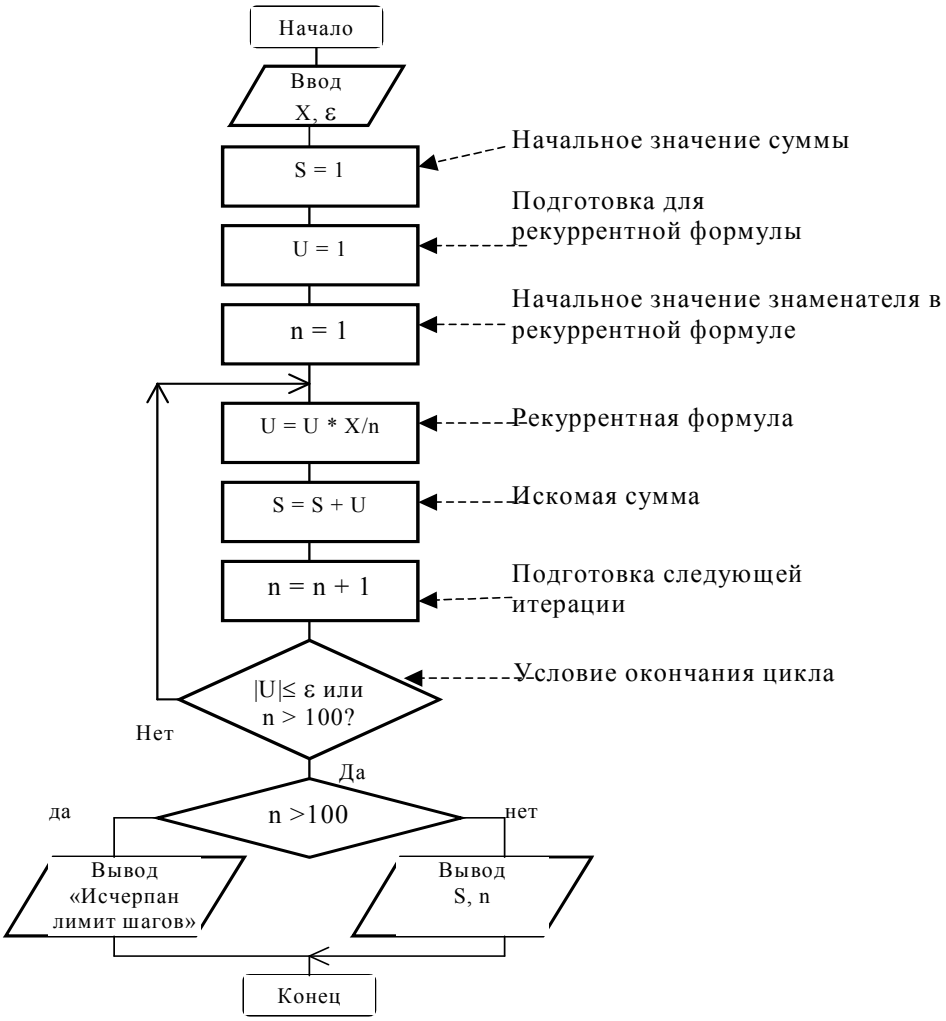



Рисунок 21. Схема алгоритма вычисления экспоненты с заданной точностью

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Iteration;
Var x, Eps, S, U: Real; n: Integer;
Const Limit = 100;          {Предельное значение шагов задано константой}
Begin
Write('Аргумент x= '); Readln(x);
Write('Погрешность Eps= '); Readln(Eps);
S := 1;                     {Частичная сумма}
U := 1;                     {Первое слагаемое}
n := 1;                     {Число шагов}
```

```
Repeat
  U := x / n * U;
  S := S + U;
  N := n + 1;
Until (abs(U) <= Eps) Or (n >= Limit);
If n>= Limit
Then Writeln(n, ' шагов не хватило для достижения заданной точности')
Else Writeln('Расчетное значение= ', S:10:2, '  число шагов= ',n);
End.
```

Результат работы программы представлен на рисунке 22.

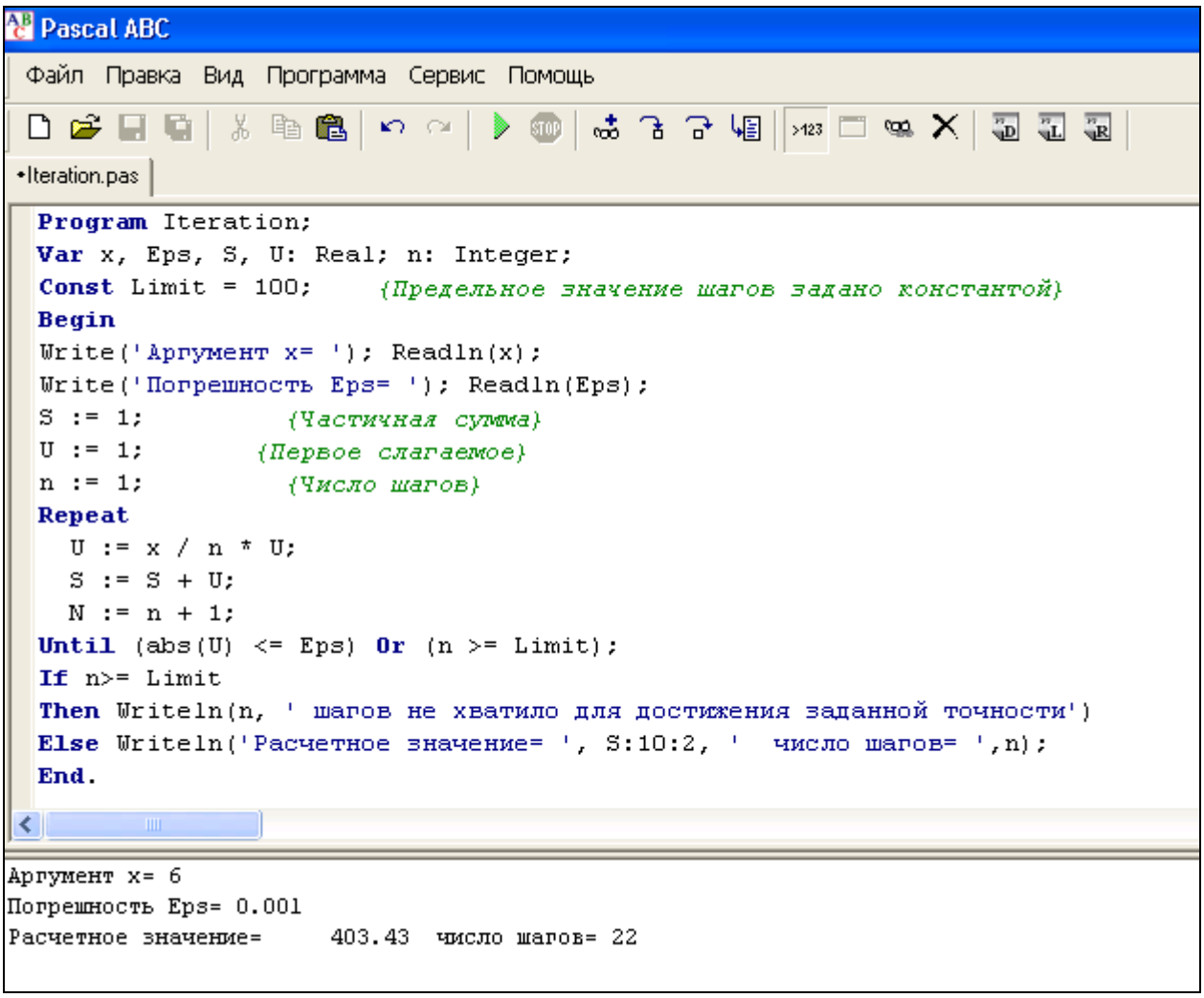



Рисунок 22. Результат работы программы вычисления экспоненты с заданной точностью

***Пример 2***

*Условие задачи*

Вычислить сумму натурального ряда чисел от 1 до N.

Программа будет состоять из трех частей, в которых повторяется решение этой задачи с использованием операторов цикла While, Repeat и For.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Natur;
Var a,N,Summa : Integer;
Begin
  Write('N='); ReadLn(N);
  {Цикл с предусловием}
  a:=1; Summa:=0;
  While a<=N Do
  Begin
     Summa:=Summa+a;
     a:=a+1;
  End;
  WriteLn('Summa=',Summa);
  {Цикл с постусловием}
  a:=1; Summa:=0;
  Repeat
     Summa:=Summa+a;
     a:=a+1;
  Until a>N;
  WriteLn('Summa=',Summa);
  {Цикл с параметром}
  Summa:=0;
  For a:=1 To N Do Summa:=Summa+a;
  WriteLn('Summa=',Summa);
End.
```

Результат работы программы представлен на рисунке 23.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу вычисления суммы бесконечного ряда $S = \frac{1}{1^3} + \frac{1}{2^3} + \frac{1}{3^3} + ... + \frac{1}{n^3} + ...$ с точностью ε=10-3. При выводе округлите результат до одного знака после запятой.

2. Составить программу, определяющую первое значение функции z = xk / k, большее *a*, если k = 1, 2, 3, …. При выводе округлите результат до одного знака после запятой.

*Вариант 2*

1. Составить программу вычисления суммы бесконечного ряда $S = x + \frac{x^3}{3} + \frac{x^5}{5} + \frac{x^7}{7} + ...$ с точностью ε=10-3. При выводе округлите результат до двух знаков после запятой.

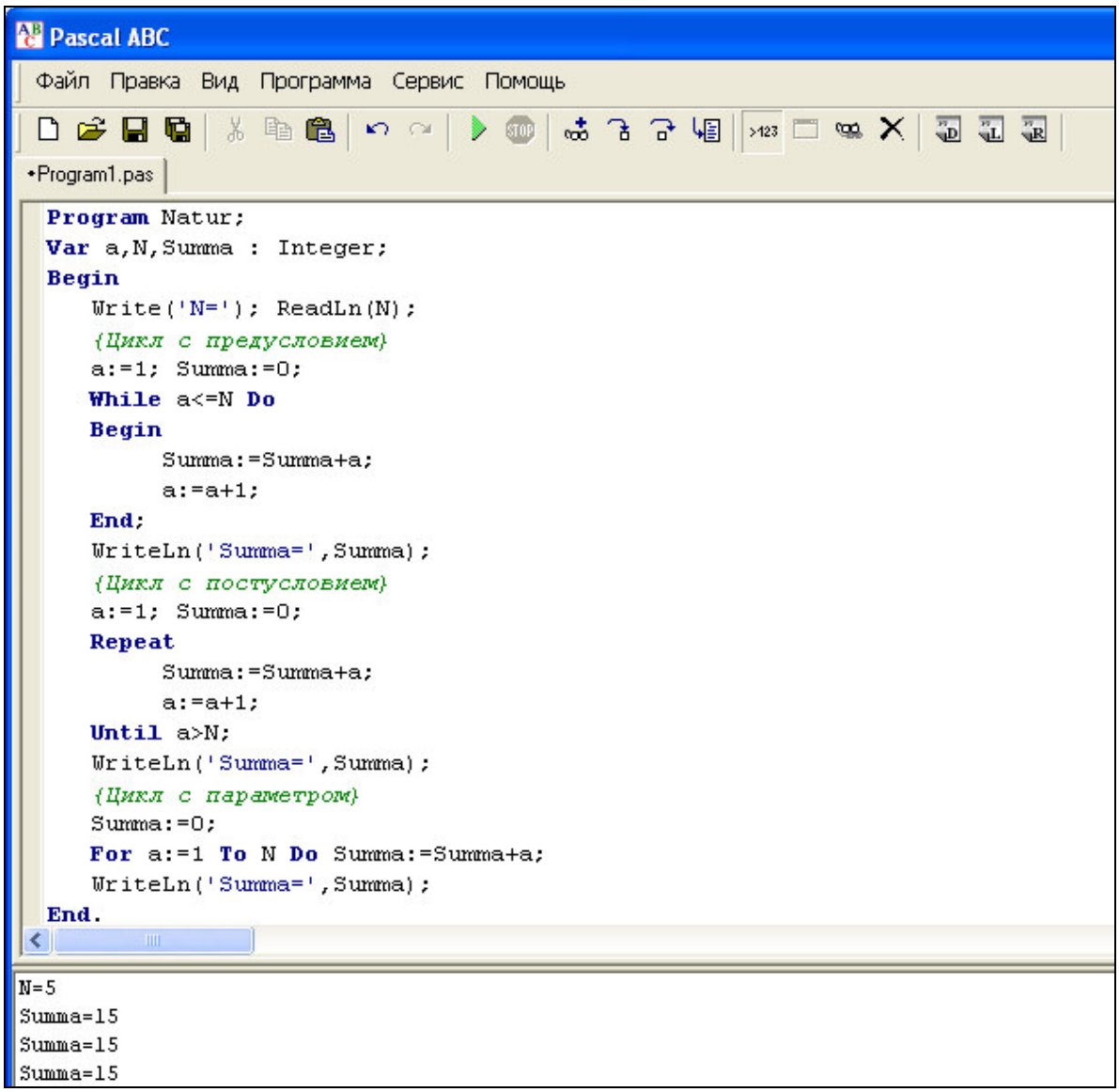



Рисунок 23. Результат работы программы, вычисляющей сумму натурального ряда

2. Составить программу, определяющую первое значение функции $z = \frac{1}{(k+1)^2}$, меньшее $a$, если k = 1, 2, 3, …. При выводе округлите результат до трех знаков после запятой.

*Вариант 3*

1. Составить программу вычисления суммы бесконечного ряда $\pi \approx 4(1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-...)$ с точностью ε=10-3. При выводе округлите результат до двух знаков после запятой.

2. Составить программу, определяющую среди чисел 1+1/2, 1+1/2+1/3, 1+1/2+1/3+1/4,… первое, большее *a*. При выводе округлите результат до двух знаков после запятой.

*Вариант 4*

1. Составить программу вычисления суммы бесконечного ряда $S=(x-1)-\frac{(x-1)^2}{2}+\frac{(x-1)^3}{3}-\frac{(x-1)^4}{4}+...$ с точностью ε=10-3. При выводе округлите результат до двух знаков после запятой.

2. Составить программу, определяющую первое значение функции $z=\frac{1}{(2k)^2}$, меньшее *a*, если k = 1, 2, 3, …. При выводе округлите результат до трех знаков после запятой.

*Вариант 5*

1. Составить программу вычисления суммы бесконечного ряда $\ln(1+x)=x-\frac{x^2}{2}+\frac{x^3}{3}-\frac{x^4}{4}+...$ с точностью ε = 10 – 3. При выводе округлите результат до двух знаков после запятой.

2. Составить программу, определяющую первое значение функции 1/x + 1/x2 + 1/x3 + + 1/x4 + … , большее *a*. При выводе округлите результат до двух знаков после запятой.

## Лабораторный практикум № 4. Программирование задач обработки одномерных и двумерных массивов

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, с основными приемами программирования при использовании переменной с индексом.

***Вводная часть***

*Одномерные массивы*

Массив – это структура данных, которую можно рассматривать как набор переменных одинакового типа, имеющих общее имя (например *a*). Каждый элемент массива имеет свой порядковый номер, называемый индексом (a[1], a[2], a[3], …, a[n]). Доступ к элементам массива осуществляется по индексу.

Массивы бывают одномерные и многомерные. Массив, в котором положение элемента определяется одним индексом, называется одномерным. В математике аналогом такой структуры является вектор – $\vec{a}$.

**Индексом может быть произвольное выражение порядкового типа, заключенное в квадратные скобки. Например Sum[True], Vect[ i + 1 ].**

Как и любая переменная в языке Паскале, массив должен быть описан.

Массивом называют как тип данных, так и переменную этого типа. Пример описания типа данных массив:

```
Type SomeArr = Array [1 .. 100] Of Real;
Var a, b, c : SomeArr;
```

Имя типа – SomeArr. Индексы в этом типе – переменные, выражения или константы целого типа. Индексы могут изменяться в интервале от 1 до 100. Каждый элемент массива имеет вещественный тип. В разделе описания переменных описаны переменные типа массив a, b, c.

Массив можно описать сразу в разделе описания переменных:

```
Var a, b, c: Array [1 .. 100] Of Real;
```

*Типовые алгоритмы обработки одномерных массивов*

**Обработка массивов производится путем изменения индексов компонент. То есть имеется способ перехода от одного элемента массива к другому путем изменения индекса. Тот факт, что индекс может быть вычисляемым объектом, выделяет массив среди многих других структур данных.**

При обработке массивов часто используется структура «цикл с параметром», так как индекс массива представляет собой счетчик и изменяется с шагом, равным 1.

Если в программе на Паскале требуется обрабатывать массивы переменной размерности, то приходится описывать массивы с максимальным возможным в данной задаче числом элементов, а затем использовать только часть из этих элементов в каждом конкретном случае.

Под вводом массива понимается получение от пользователя во время работы программы значений элементов массива.

Фрагмент ввода массива:

```
For i := 1 To n Do
Begin
   Write('Введите a[', i, ']'); Readln(a[i]);
End;
```

Ввод массива необходимо производить поэлементно. Индекс элемента массива должен пробежать все значения от 1 до n. Для этого необходимо организовать цикл, в котором индекс массива является параметром цикла. Оператору ввода предшествует оператор вывода, с помощью которого осуществляется вывод подсказки.

Под выводом массива понимается вывод всех значений элементов массива.

Фрагмент вывода массива:

```
For i := 1 To n Do
Begin
   Writeln('a[' , i, ']=', a[i]);
End;
```

Для расчета суммы элементов одномерного массива используется прием накопления суммы:

```
S := 0;
For i := 1 To n Do S := S + a[i];
```

Этот прием заключается в том, что в формуле вычисления суммы в правой и левой частях выражения записано имя одной и той же переменной. Начальное значение должно быть равно 0. В цикле к текущему значению суммы S прибавляется значение очередного элемента массива, и результат записывается в переменную S, которая для следующего прохода цикла становится текущим значением. По окончании работы оператора цикла в переменной S накопится сумма всех элементов массива.

Для определения среднего значения элементов массива полученную сумму следует разделить на количество элементов массива и вывести результат на экран.

*Двухмерные массивы*

Массив – это структура данных, которую можно рассматривать как набор переменных одинакового типа, имеющих общее имя (например *a*). Массивы бывают одномерные и многомерные.

В языке Паскаль имеется возможность работы с двухмерными, трехмерными массивами, а также массивами большей размерности. Наиболее часто используются двухмерные массивы.

В математике аналогом двухмерного массива является матрица.

$$A = \begin{bmatrix} 1 & 1 & 0 & 3 \\ 3 & 3 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 2 & 2 & 3 & 3 \end{bmatrix}.$$

Матрица A состоит из четырех строк и четырех столбцов. В матрице каждый элемент идентифицируется номером строки и номером столбца, на пересечении которых он расположен.

В двухмерном массиве элемент также определяется двумя индексами, которые заключаются в квадратные скобки и отделяются друг от друга запятыми, например

A[2, 3].

В двухмерном массиве первый индекс указывает на номер строки, а второй индекс указывает на номер столбца, на пересечении которых расположен элемент. В матрице A, записанной выше, элемент A[2, 3] равен 1, а элемент A[3, 2] равен 0.

Как и любая переменная в языке Паскале, массив должен быть описан.

Формат записи двухмерного массива в языке Паскаль:

<имя>: Array [<нижний_индекс1>..<верхний_индекс1>,
<нижний_индекс2>..<верхний_индекс2>] Of <тип>;

Пример описания типа данных двухмерный массив:

```
Type SomeArr = Array [1 .. 10, 1 .. 20] Of Real;
Var a, b, c : SomeArr;
```

Имя типа – SomeArr. Первый индекс может изменяться в интервале от 1 до 10, второй индекс может изменяться в интервале от 1 до 20. Каждый элемент массива имеет вещественный тип. В разделе описания переменных описаны переменные типа массив a, b, c.

Массив можно описать сразу в разделе описания переменных:

```
Var a, b, c: Array [1 .. 10, 1 .. 20] Of Real;
```

*Типовые алгоритмы обработки двухмерных массивов*

**Обработка массивов производится путем изменения индексов компонент. То есть имеется способ перехода от одного элемента массива к другому путем изменения индексов.**

При обработке массивов часто используется структура «цикл с параметром», так как индекс массива представляет собой счетчик и изменяется с шагом, равным 1.

Если необходимо обнулить все элементы первой строки, то можно рассматривать строку матрицы как одномерный массив. Тогда нужно записать оператор:

```
For i:=1 To n Do a [1, i] := 0;
```

Первый индекс элемента остается неизменным, он равен 1. Это номер строки. Второй индекс изменяется от 1 до n. Это номер столбца.

Если необходимо обнулить все элементы первого столбца, то можно рассматривать столбец матрицы как одномерный массив. Тогда нужно записать оператор:

```
For i:=1 To n Do a [i ,1] := 0;
```

В цикле организуем изменение первого индекса. Второй индекс, определяющий номер столбца, остается неизменным.

**Если нужно обнулить все элементы матрицы, то следует организовать два вложенных цикла:**

```
For i := 1 To n Do
   For j := 1 To m Do
         a[i,j] := 0;
```

При каждом значении i (номера строки) j пробегает все значения от 1 до m.

Под вводом массива понимается получение от пользователя во время работы программы значений элементов массива.

Фрагмент ввода двухмерного массива:

```
For i := 1 To n Do
  For j := 1 To m Do
  Begin
    Write('Введите a[', i, ',' j, ']='); Readln(a[i,j]);
  End;
```

Ввод двухмерного массива необходимо производить поэлементно. Для этого необходимо организовать два цикла по строкам и по столбцам для определения индексов элементов массива. Оператору ввода предшествует оператор вывода, с помощью которого осуществляется вывод подсказки.

Под выводом массива понимается вывод всех значений элементов массива.

Фрагмент вывода двухмерного массива в виде таблицы:

```
For i := 1 To n Do
Begin
  For j := 1 To m Do
    Write(a[i,j]);
  Writeln;
End;
```

Так же как и при вводе массива, следует организовать два цикла по строкам и по столбцам для определения индексов элементов массива. Элементы массива A выводятся оператором Write. Это означает, что курсор после очередного вывода остается в той же строке. Элементы массива выводятся в той же строке до тех пор, пока работает цикл по j. Как только цикл по j завершится, пустой оператор Writeln переведет курсор на новую строку. Такая организация вывода позволяет представить информацию в виде матрицы.

Для расчета суммы элементов двухмерного массива используется прием накопления суммы:

```
S := 0;
For i := 1 To n Do
    For j := 1 To m Do
S := S + a[i,j];
```

Этот прием заключается в том, что в формуле вычисления суммы в правой и левой частях выражения записано имя одной и той же переменной. Начальное значение должно быть равно 0. В цикле к текущему значению суммы S прибавляется значение очередного элемента массива, и результат записывается в переменную S, которая для следующего прохода цикла становится текущим значением. По окончании работы операторов цикла в переменной S накопится сумма всех элементов массива.

Для определения среднего значения элементов массива полученную сумму следует разделить на количество элементов массива и вывести результат на экран.

Для расчета произведения элементов двухмерного массива используется прием накопления произведения:

```
P := 1;
For i := 1 To n Do
    For j := 1 To m Do
P := P * a[i,j];
```

Начальное значение произведения должно быть равно 1.

***Практическая часть***

*Пример 1*

Написать программу, которая вычисляет среднее арифметическое значений элементов одномерного массива.

Схема алгоритма приведена на рисунке 21.

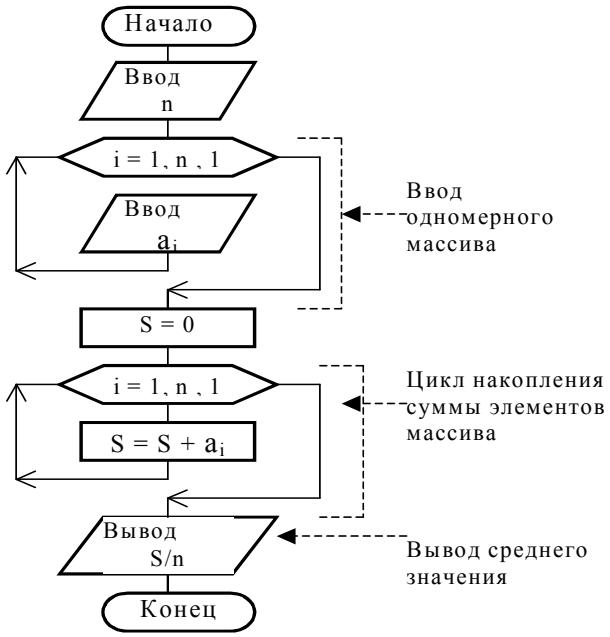



Рисунок 21. Схема алгоритма, вычисляющего среднее арифметическое значение элементов одномерного массива

Программа на Паскале в этом случае может иметь следующий вид:

```
Program mid;
{Программа вычисляет среднее арифметическое значений
элементов массива}
Const nn = 10;
Var
   a: Array[1..nn] Of Integer;
   i, n: Integer;
   S: Real;
Begin
{Ввод количества элементов массива}
   Write('Количество элементов? '); Readln(n);
{Ввод элементов массива}
   For i := 1 To n Do
   Begin
            Write('Введите a[' , i, ']='); Readln(a[i]);
   End;
{Вычисление суммы элементов массива}
S := 0;
   For i := 1 To n Do S := S + a[i];
{Вывод результата на экран}
   Writeln('Среднее арифметическое значение элементов массива: ', (S/n):5:3);
End.
```

В разделе констант определена константа nn, значение которой используется для определения граничного индекса. Использование раздела констант позволяет сгруппировать в начале программы величины, характерные для конкретного примера.

Массив *a* описан в разделе описания переменных. Это массив целых чисел. Индекс элементов массивы изменяется в интервале от 1 до nn.

Переменная i используется в качестве параметра оператора цикла. Переменная n – количество элементов массива, исходная величина. S – результат вычислений. В этой переменной накапливается сумма элементов массива.

Результат работы программы представлен на рисунке 22.

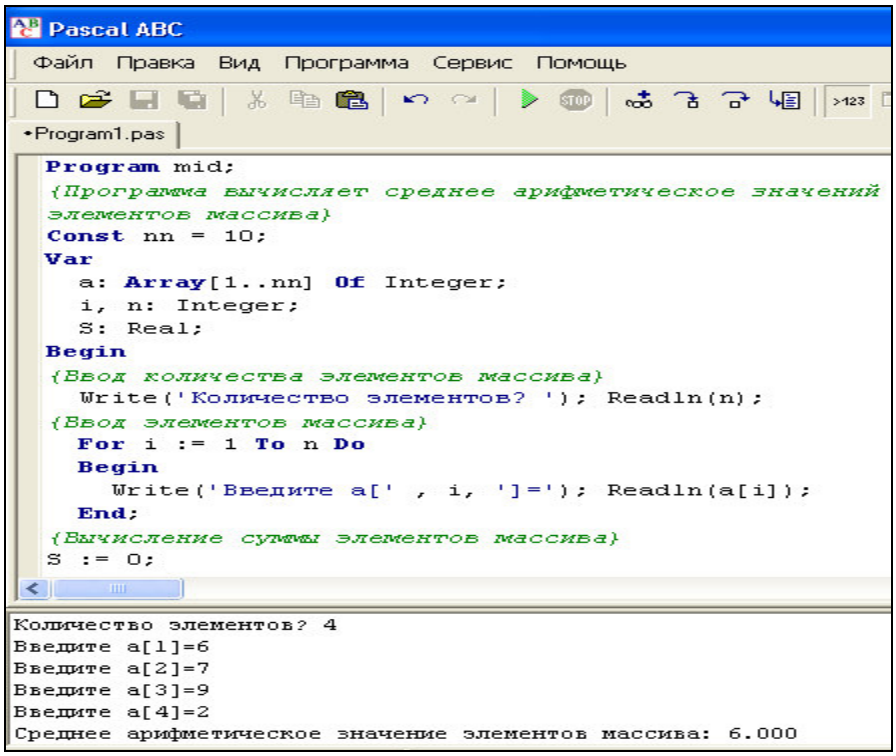



Рисунок 22. Результат работы программы, вычисляющей среднее арифметическое значение элементов одномерного массива

***Пример 2***

*Условие задачи*

Найти максимальный элемент последовательности, состоящей из n элементов.

Фрагмент схемы алгоритма приведен на рисунке 23.

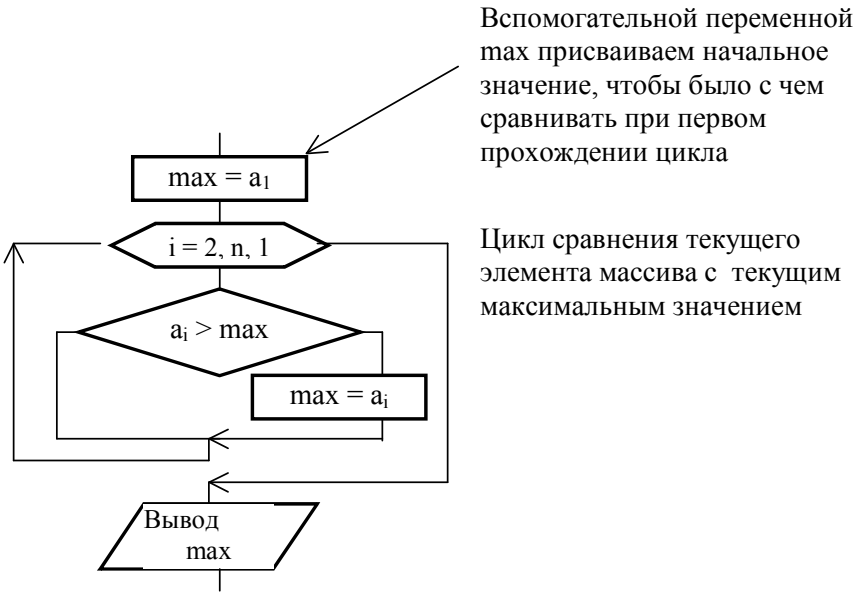



Рисунок 23. Фрагмент алгоритма, осуществляющего поиск максимального элемента последовательности

Программа на Паскале в этом случае может иметь следующий вид:

```
Program p_max;                              {Поиск максимального элемента}
Const nn = 100;
Type list = 1..nn;
Var i : list;
   max, n : Integer;
   a : Array [list] Of Integer;
Begin
Write('Количество элементов? '); Readln(n); {Ввод количества элементов массива}
For i := 1 To n Do                          {Ввод элементов массива}
Begin
  Write('Введите ',i,'-й элемент массива: '); Readln(a[i])
end;
max := a[1];
For i:= 2 To n Do
  If a[i] > max Then max := a[i];
Write ('Максимальный элемент массива: ', max)
end.
```

Тип данных list, заданный в разделе описания типов, используется в качестве граничной пары в описании массива. Вспомогательная переменная i, которая используется в качестве параметра цикла, имеет такой же тип, как и индекс массива. Это обеспечивает дополнительный контроль соответствия параметра цикла индексу массива.

Переменная max принимает значение первого элемента массива *a*. Над всеми остальными элементами массива выполняется однотипная операция: элемент массива сравнивается с переменной max. Если встречается элемент массива больший max, то max приобретает значение этого элемента. Если условие ai > max не выполняется, то присваивания не происходит. По окончании цикла в переменной max останется значение самого большого элемента.

Результат работы программы представлен на рисунке 24.

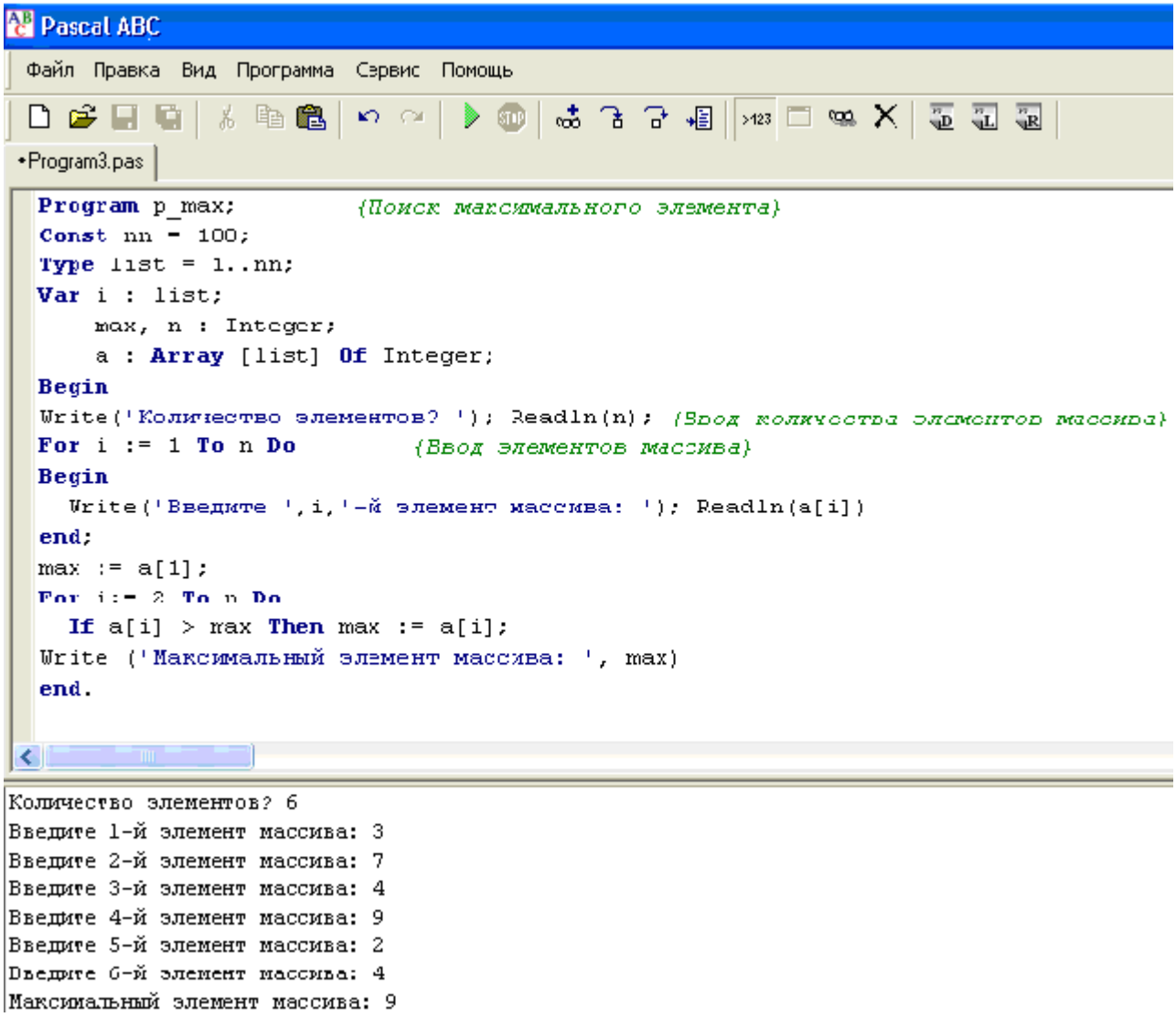



Рисунок 24. Результат работы программы, осуществляющей поиск максимального элемента последовательности

***Пример 3***

*Условие задачи*

Разработать алгоритм и написать программу сортировки одномерного массива по возрастанию. Для решения задачи воспользуемся алгоритмом, который получил название «метод пузырька». Этот алгоритм заключается в следующем. Перебираются все элементы массива и производятся сравнения: первый элемент со вторым, второй с третьим и так далее до n-го. Если значение некоторого элемента больше значения следующего элемента, то эти элементы меняются местами. В результате последовательность может быть не отсортирована. Однако после первого прохода можно с уверенностью сказать, что наибольшее значение находится на своем месте – на правой границе массива. Выполнив еще один проход по n – 1 элементам, получим два упорядоченных значения. Если осуществить n – 1 проход, получим упорядоченный массив.

Фрагмент схемы алгоритма представлен на рисунке 25.

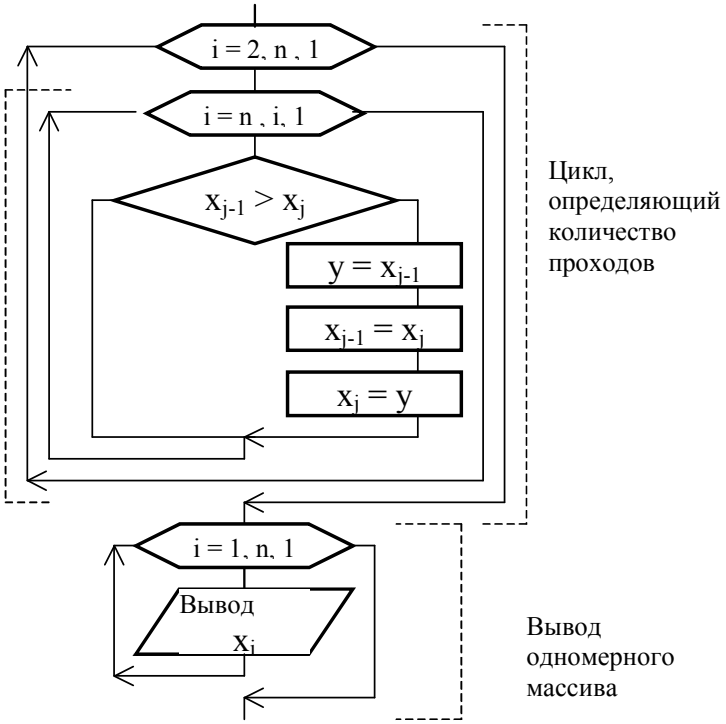



Рисунок 25. Фрагмент алгоритма, осуществляющего сортировку одномерного массива

Программа на Паскале в этом случае может иметь следующий вид:

```
Program puzirok;
{Упорядочивание элементов массива методом "пузырька"}
Const nn = 100;
Type mass = Array [1 .. nn] Of Real;
Var i, j, n : Integer;
   y : Real;
   x : mass;
Begin
{Ввод исходной информации}
Write('Количество элементов? '); Read(n);
For i := 1 To n Do
Begin
  Write('Введите ',i,'-й элемент массива: '); Readln(x[i])
End;
{Упорядочивание массива по возрастанию}
```

```
For i := 2 To n Do                                    {Проход}
Begin
  For j := n Downto i Do                              {Цикл для сравнения}
    If x[j - 1] > x[j] Then
    Begin
      y := x[j - 1]; x[j - 1] := x[j]; x[j] := y      {Перестановка}
    End;
End;
{Вывод результата - упорядоченного массива}
Writeln('Упорядоченный массив:');
For i:= 1 To n Do Write(x[i]:8:2)
End.
```

Этот алгоритм реализуется с помощью двух циклов For – внутреннего и внешнего. Число повторений внешнего цикла на 1 меньше числа элементов массива (так как последовательность из одного значения проверять нет смысла). Число повторений внутреннего цикла последовательно сокращаться от n до 1.

Внешний цикл по i определяет количество проходов. Цикл по j создан для сравнения пар элементов массива. Сравниваются два элемента, расположенных рядом. Если XJ-1 > XJ, то осуществляется перестановка элементов местами. Для того чтобы не потерять значение одного из переставляемых элементов, вводится вспомогательная переменная Y, которая представляет собой буфер для хранения x[j-1].

Для вывода массива, так же как и для ввода, требуется организовать цикл, определяющий индекс каждого элемента массива. Вывести массив – значит вывести все его элементы. Выводится Xi, при этом i пробегает все значения от 1 до n с шагом 1.

Результат работы программы представлен на рисунке 26.

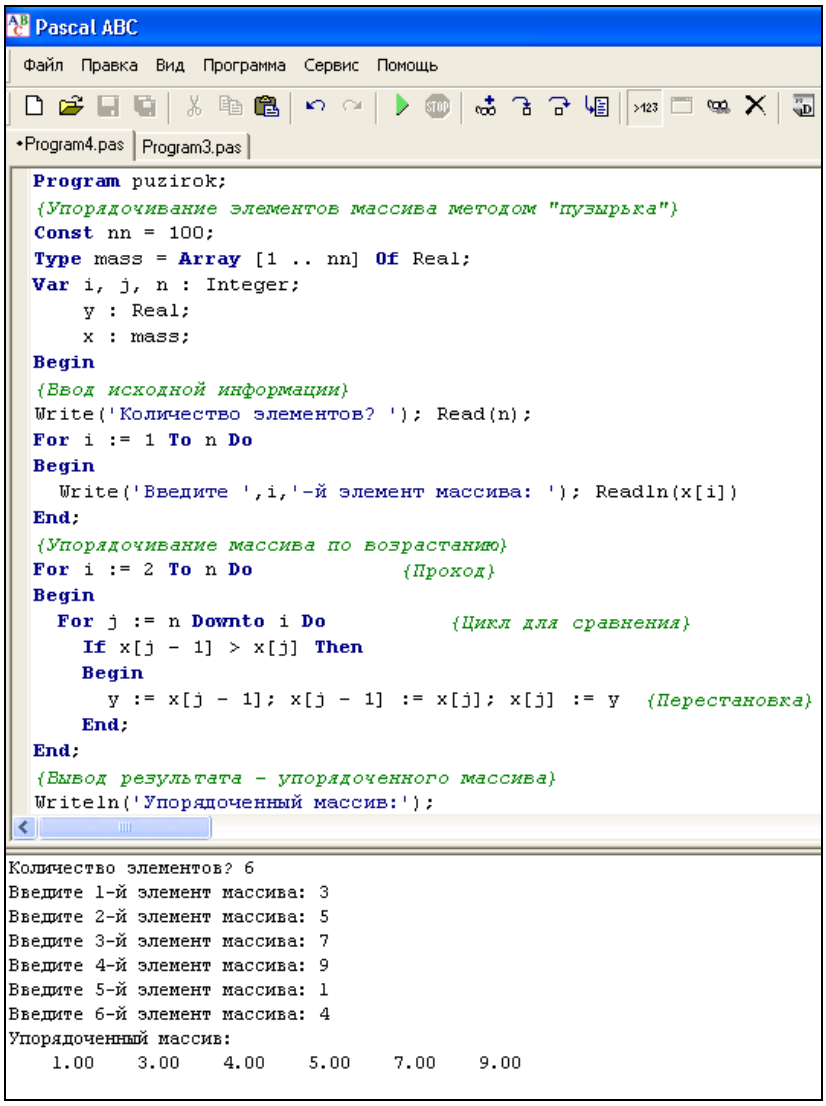



Рисунок 26. Результат работы программы, осуществляющей сортировку одномерного массива

***Пример 4***

*Условие задачи*

Рассчитать суммы элементов строк матрицы. Результат представить в виде одномерного массива.

Дано: n – количество строк матрицы, m – количество столбцов матрицы.

[B] – матрица с элементами bij, где i изменяется от 1 до n, j изменяется от 1 до m.

Найти: Sumi = $\sum_{j=1}^{m} b_{ij}$ .

Фрагмент схемы алгоритма показан на рисунке 27.

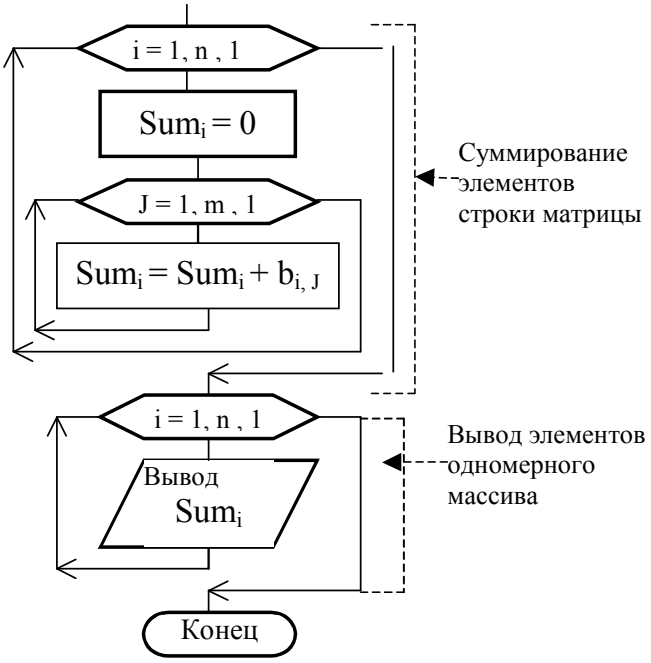



Рисунок 27. Фрагмент алгоритма, вычисляющего сумму элементов строк матрицы

Программа на Паскале в этом случае может иметь следующий вид:

```
Program SumStrok;
Type Mass = Array [1..10] Of Real;
Var i, j, n, m: Integer;
    Sum: Mass;
    b: Array [1..10,1..10] Of Real; {Описали массив}
Begin
Write('Количество строк матрицы? '); Readln(n);
Write('Количество столбцов матрицы? '); Readln(m);
For i:=1 To n Do
For j:=1 To m Do
Begin
  Write('Введите b[' , i, ',', j,']='); Readln(b[i,j]);
End;
For i:=1 To n Do
Begin
  Sum[i]:=0;
  For j:=1 To m Do Sum[i] := Sum[i] + b[i,j];
End;
For i:= 1 To n Do WriteLn ('Сумма элементов строки ', i, ' = ',Sum[i] : 10:3);
End.
```

Сумма каждой строки матрицы должна храниться в соответствующем элементе одномерного массива. Так, сумма 1-й строки должна храниться в первом элементе Sum1, сумма 2-й строки – в элементе Sum2 и так далее. Цикл по i определяет номер элемента одномерного массива Sum и номер строки матрицы B. Индекс j определяет номер строки матрицы. В цикле по j накапливается сумма элементов соответствующей строки матрицы.

На поиск в массиве элемента с заданными значениями индексов затрачивается время. (Адрес i-го элемента определяется прибавлением к адресу начала массива значения i.) Поэтому для повышения эффективности

лучше использовать вспомогательную переменную S при суммировании, что исключает многократное обращение к элементам массива Sum.

Результат работы программы представлен на рисунке 28.

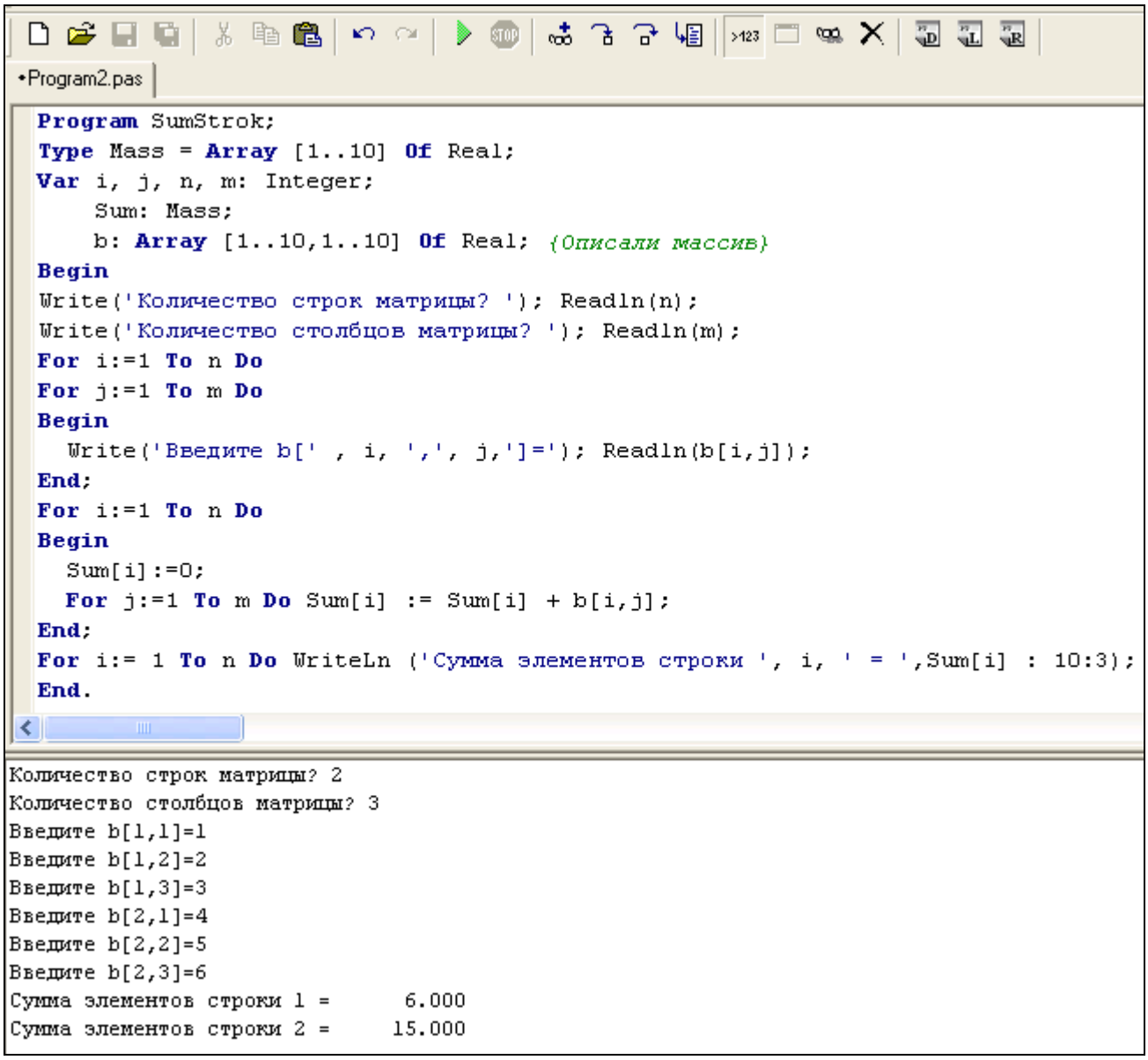



Рисунок 28. Результат работы программы, вычисляющей сумму элементов строк матрицы

***Пример 5***

*Условие задачи*

Дана матрица размером N на M. Поменять местами четные и нечетные столбцы матрицы.

Фрагмент схемы алгоритма показан на рисунке 29.

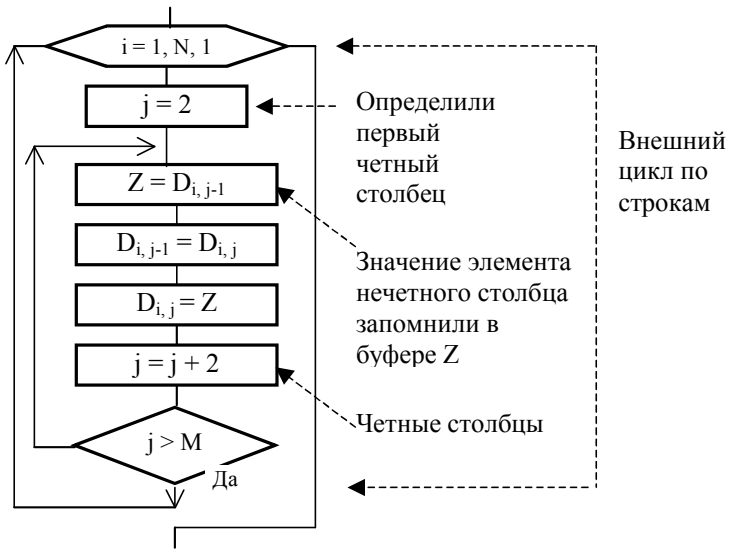



Рисунок 29. Фрагмент алгоритма, меняющего местами четные и нечетные столбцы матрицы

Для перестановки элементов четных и нечетных столбцов организован цикл с шагом 2. Внутри цикла по строкам организован цикл структуры «До». Перебираются только четные столбцы (со второго по последний с шагом 2). Нечетный столбец определен как J – 1. Во избежание потери информации вводится вспомогательная переменная Z, которая является буфером для хранения значения элемента нечетного столбца.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Change_Stolb;
Var i, j, n, m: Integer;
    Z: Real;
    D: Array [1..10, 1..10] Of Real;
Begin
{Ввод размера матрицы}
Write('Количество строк матрицы? '); Readln(n);
Write('Количество столбцов матрицы? '); Readln(m);
{Ввод матрицы}
For i:=1 To n Do
For j:=1 To m Do
Begin
  Write('Введите D[', i, ',', j,']='); Readln(D[i,j]);
End;
{ ***Перестановка столбцов ***}
For i:=1 To n Do              {Цикл по строкам}
Begin
  j:=2;                 {Подготовка цикла по столбцам}
  Repeat                {Цикл по столбцам}
  {Перестановка четного и нечетного элементов}
    Z:=D[i,j-1]; D[i,j-1]:=D[i,j]; D[i,j]:=Z;
    j:=j+2;             {Перебираем только четные столбцы}
  Until j>m;
End;
{Вывод результирующей матрицы}
For i:=1 To n Do
Begin
  For j:=1 To m Do
```

```
    Write(D[i,j]:8:3);
  Writeln;
End;
End.
```

Результат работы программы представлен на рисунке 30.

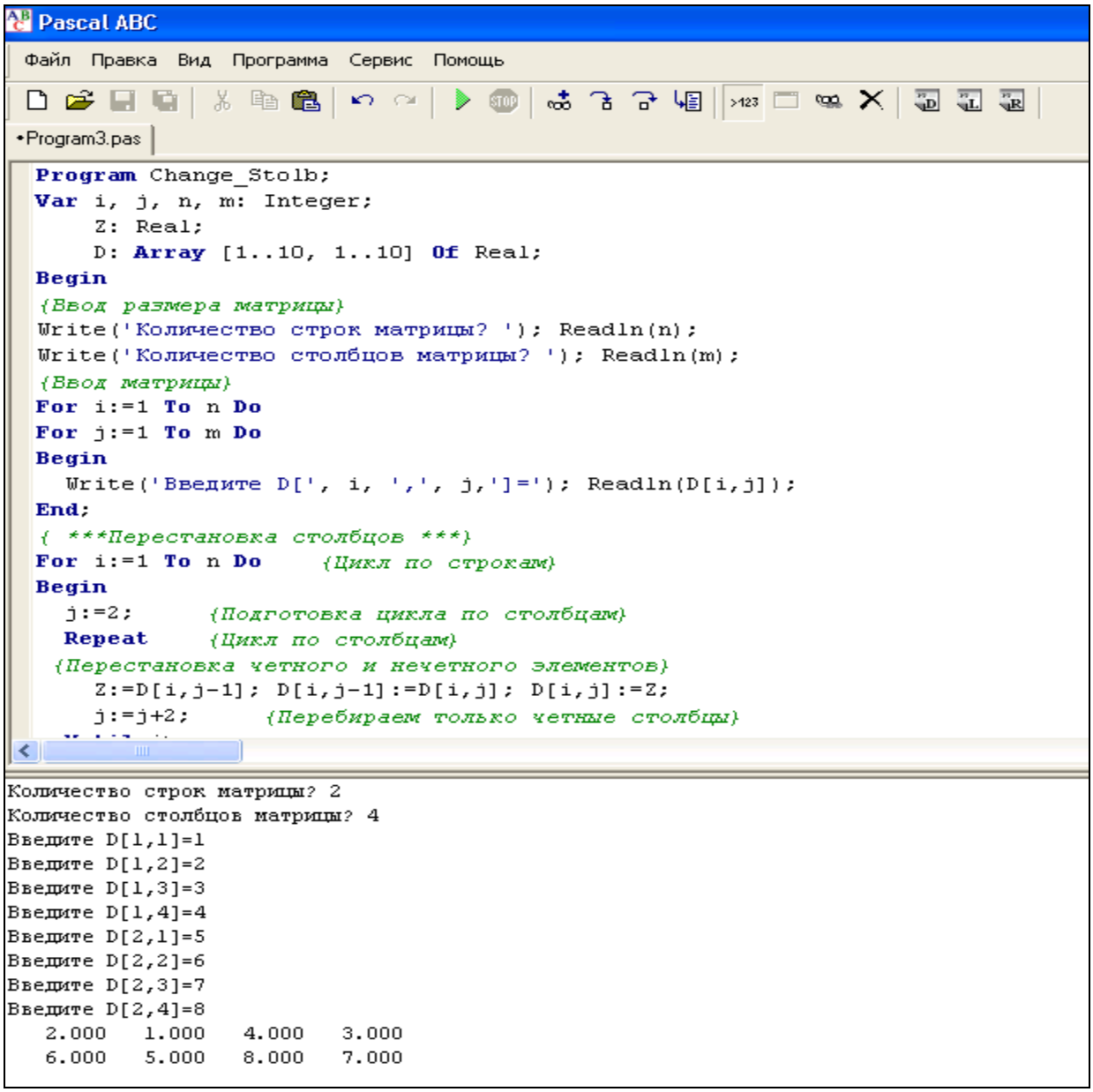



Рисунок 30. Результат работы программы, меняющей местами четные и нечетные столбцы матрицы

***Самостоятельная работа***

*Вариант 1*

1. Составить программу, заполняющую одномерный массив A тридцатью первыми членами арифметической прогрессии (первый член прогрессии равен 10, ее разность равна 5): $a_1 = 10; a_{i+1} = a_i + 5$. Определить сумму элементов массива A.

2. Составить программу, формирующую массив B из массива A (см. задачу 1), при этом все элементы, превышающие заданное число Z, заменяются этим числом. Подсчитать количество выполненных замен. Определить произведение первых четырех элементов полученного одномерного массива B.

3. Определить количество элементов матрицы A, значение которых превышает заданное число.

4. Найти среднее арифметическое каждого из столбцов матрицы A. При выводе округлите результаты до трех знаков после запятой.

*Вариант 2*

1. Составить программу, заполняющую одномерный массив A пятнадцатью первыми членами геометрической прогрессии (первый член прогрессии равен 1, ее знаменатель равен 2): $a_1 = 1; a_{i+1} = a_i * 2$. Определить сумму элементов массива A.

2. Составить программу, формирующую массив B из массива A (см. задачу 1): $b_i = \frac{a_i}{2}$. Определить среднее арифметическое всех элементов и произведение первых пяти элементов полученного одномерного массива B. При выводе округлите результаты до двух знаков после запятой.

3. Определить количество отрицательных и положительных элементов матрицы A.

4. Найти среднее арифметическое каждой из строк матрицы A. При выводе округлите результаты до двух знаков после запятой.

*Вариант 3*

1. Составить программу, заполняющую одномерный массив A двадцатью первыми членами последовательности Фибоначчи (последовательности, в которой первые два члена равны 1, а каждый следующий равен сумме двух предыдущих: $a_1 = a_2 = 1; a_i = a_{i-1} + a_{i-2}$). Определить сумму элементов массива A и произведение первых семи элементов массива.

2. Составить программу, формирующую массив B из массива A (см. задачу 1): $b_i = \sqrt{a_i}$. Определить количество элементов полученного одномерного массива B, значения которых принадлежат интервалу [9;30], и среднее арифметическое всех элементов массива B. При выводе округлите результат до двух знаков после запятой.

3. Определить количество отрицательных и положительных элементов матрицы A.

4. Найти сумму элементов каждого из столбцов матрицы A. При выводе округлите результаты до двух знаков после запятой.

*Вариант 4*

1. Составить программу, заполняющую одномерный массив A натуральными числами, делящимися нацело на 13 и лежащими в интервале [20;300]. Определить количество и сумму элементов одномерного массива A.

2. Составить программу, формирующую массив B из массива A (см. задачу 1): $b_i = 2a_i$. Определить среднее арифметическое всех элементов массива B и произведение первых трех элементов массива B.

3. Определить сумму и среднее арифметическое элементов матрицы A, лежащих на главной диагонали.

4. Найти сумму элементов каждого из столбцов матрицы A.

*Вариант 5*

1. Составить программу, заполняющую одномерный массив A: $a_i = \frac{c+d}{i}, i = \overline{1,20}, c = 20, d = 30$. Определить сумму элементов массива A. При выводе округлите результат до двух знаков после запятой.

2. Составить программу, формирующую массив B из массива A (см. задачу 1): $b_i = \frac{\sqrt{a_i}}{i^2}$. Определить количество элементов полученного одномерного массива B, значения которых меньше 0.1, и среднее арифметическое всех элементов массива B. При выводе округлите результат до двух знаков после запятой.

3. Определить сумму элементов матрицы A, лежащих выше и ниже главной диагонали. При выводе округлите результаты до двух знаков после запятой.

4. Найти сумму элементов каждой из строки матрицы A. При выводе округлите результаты до двух знаков после запятой.

## Лабораторный практикум № 5. Программирование задач обработки строковых данных

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение приемов программирования при работе со строковыми данными в языке Паскаль.

***Вводная часть***

*Строковый тип и действия, которые можно производить с переменными строкового типа*

Для обработки текстов в языке Паскаль используется строковый тип.

Строковая константа широко использовалась в комментариях для списков вывода оператора Write. Строковая константа – это совокупность любых символов, заключенная в апострофы, например: 'Количество отрицательных элементов вектора =', 'Введите n', 'Нажмите <Enter>' и т.д.

Переменная строкового типа описывается с помощью служебного слова String, например:

```
Var
S: String;
Str_simv: String[20];
```

В описании типа String длина строки может принимать любое целое значение от 1 до 255. Если длина не указана, то берется максимальная длина, равная 255. Переменная Str_simv может хранить строки длиной не

более 20 символов. Описание String без указания длины отводит для переменной место, как и описание String[255].

Можно описать строковый тип, а затем этот тип использовать для описания переменных:

```
Const N=10;
Type
Str = String[20]; Str_1 = String[N];
Var
Value_a, Property_b: Str;
. . .
```

Для переменной типа String[N] выделяется N + 1 байт памяти.

S[0] – хранит код, соответствующий числу символов в строке.

Ord(S[0]) – длина строки S.

Значением строки может быть любая последовательность символов, заключенная в одинарные кавычки:

'abcde-абвгд'

'123 ? 321'

Строки можно присваивать, объединять и сравнивать.

```
S := 'Кон' +'кат' + 'ена' + 'ция';
```

Сложение строк производится с помощью оператора конкатенации. Этот оператор записывается с помощью знака +. При выполнении действия сложения начало строки, идущей после знака “+”, подсоединяется к концу строки, указанной до этого знака.

Произвольные пары строк могут сравниваться с помощью операторов отношений: = , <> , < , > , <=, >= :

'Abcd' < 'abcd'

Сравнение происходит посимвольно слева направо. При этом символ 1 считается меньше символа 2, если его код в таблице ASCII меньше, чем у символа 2. Код заглавной буквы «А» меньше, чем у прописной буквы «а». Результат сравнения этих двух строк (Abcd с заглавной буквой «А» и abcd с прописной “a”) есть TRUE (истина).

Обработка текстов имеет некоторое сходство с обработкой массивов. Символы в тексте (как и элементы в массиве) упорядочены по номеру позиции, которую они занимают, и переход к следующему символу легко осуществляется изменением номера позиции на 1. Поэтому многие типовые алгоритмы обработки массивов в несколько модифицированном виде могут использоваться и при обработке текстов.

К любому символу в строке можно обратиться так же, как к элементу одномерного массива. Элементы строки идентифицируются именем строки с индексом, заключенным в квадратные скобки. Например: N[5], S[i], Slovo[k + 1]. Первый номер строки имеет номер 1. Значение индекса не должно выходить за границы описания.

*Стандартные процедуры и функции, используемые при работе со строками*

При работе со строками используется набор стандартных процедур и функций (таблица 1).

Таблица 1. Набор стандартных процедур и функций для работы со строками

| Процедура/функция | Действие | Пример |
|---|---|---|
| Length(S : String): Integer; | Возвращает текущую длину строки S (число символов, из которых состоит строка S) | Stroka := 'Пример';<br>L := Length(Stroka);<br>L равно 6 |
| Copy(S : String; Start, Len: Integer) : String; | Результат – строковая величина, состоящая из Len символов строки S, начиная с символа номер Start | Str := Copy('Function', 2, 4);<br>Результат: 'unct'.<br>Str := Copy('Сарай', 3, 3);<br>Результат: 'рай' |
| Pos(Subs, S : String) : Integer; | Ищет вхождение подстроки Subs в строке S и возвращает номер первого | i := Pos('о', 'Космодром');<br>Результат: 2. |

| Процедура/функция | Действие | Пример |
|---|---|---|
| | символа Subs в S или 0 , если S не содержит Subs. | p := Pos('одр', 'Космодром');<br>Результат: 5 |
| Insert(Subs : String; Var S : String; Start : Integer); | Вставляет подстроку Subs в строку S, начиная с позиции Start | S := 'Начало–конец';<br>Insert(S, 'середина–', 8);<br>Результат: 'Начало–середина–конец' |
| Delete(Var S : String; Start, Len : Integer); | Видоизменяет строку S, удаляя Len символов, начиная с номера Start | S := 'Строка';<br>Delete(S, 2, 4);<br>Результат: ‘Са’. |
| Str(X [ : Width [ : Dec] ]; Var S : String); | Преобразует численную величину X в последовательность символов, соответствующей значению величины, затем преобразует эту последовательность в строковую величину, которую присваивает переменной S | Str(45, S);<br>Результат: '45'.<br>Str(45:3, S);<br>Результат: '{45' |
| Val(S: String; Var V; Var ErrCode : Integer); | Преобразует строковое выражение S в число и присваивает его значение переменной V. Если преобразование возможно, то переменная ErrCode равна нулю, в противном случае она содержит номер символа в S, на котором процедура застопорилась. Тип V должен соответствовать содержимому строки S. Если в S имеется точка, то V должна быть вещественного типа | Пусть Var IntVar, Rep : Integer;<br>Процедура:<br>Val('23', IntVar, Rep);<br>IntVar = 23.<br>Rep = 0<br>Процедура:<br>Val('23.0', IntVar, Rep);<br>IntVar = неизменно.<br>Rep = 3 |
| Concat(S1,S2,...SN : String): String; | Возвращает строку, являющуюся результатом слияния строк `s1`,..., `sn`. Результат тот же, что у выражения `s1+s2+...+sn` | Stroka1:='Комп';<br>Stroka2:='ьютер';<br>Stroka:= Stroka:=Concat(Stroka1, Stroka2);<br>Stroka равна 'Компьютер' |
| `IntToStr(I:Integer):String;` | Преобразует целое число к строке | I:=6;<br>Str:=IntToStr(I);<br>Str равна '6' |
| `StrToInt(S:String):Integer;` | Преобразует строку в целое число. Если преобразование невозможно, то возникает ошибка времени выполнения | Str:='6';<br>I:=StrToInt(Str);<br>I равно 6 |
| FloatToStr(V:Real):String; | Преобразует вещественное число к строке | V:=6.5;<br>Str:=FloatToStr(I);<br>Str равна '6.5' |
| StrToFloat(S:String):Real; | Преобразует строку в вещественное число. Если преобразование невозможно, то возникает ошибка времени выполнения | Str:='6.5';<br>V:=StrToFloat(Str);<br>V равно 6.5 |
| `UpCase(C:Char):Char;` | Возвращает символ `C`, преобразованный к верхнему регистру | C:=UpCase('a');<br>Результат: 'A' |
| `LowCase(C:Char):Char;` | Возвращает символ `C`, преобразованный к нижнему регистру | C:=LowCase('A');<br>Результат: 'a' |
| `UpperCase(S:String):String;` | Возвращает строку `S`, преобразованную к верхнему регистру | S:=UpperCase('abcd');<br>Результат: 'ABCD' |
| `LowerCase(S:String):String;` | Возвращает строку S, преобразованную к нижнему регистру | S:=UpperCase('ABCD');<br>Результат: 'abcd' |
| `Trim(S:String):String;` | Возвращает копию строки S с удаленными лидирующими и заключительными пробелами | S:=Trim('  Abcd ');<br>Результат: 'Abcd' |
| `TrimLeft(S:String):String;` | Возвращает копию строки `S` с удаленными лидирующими пробелами | S:=TrimLeft('  Abcd ');<br>Результат: 'Abcd ' |
| `TrimRight(S:String):String;` | Возвращает копию строки `S` с | S:=TrimRight('  Abcd ');<br>Результат: ' Abcd' |

| Процедура/функция | Действие | Пример |
|---|---|---|
| | удаленными заключительными пробелами | |

***Практическая часть***

***Пример 1***

*Условие задачи*

Выделить и напечатать первое слово текста.

Программа на Паскале в этом случае может иметь следующий вид:

```
Var V, T : String; P1 : integer;
Begin
Write('Введите текст: '); Readln(T);
P1 := Pos(' ', T);
V := Copy(T, 1, P1 - 1);
Writeln('Первое слово введенного текста: ', V);
End.
```

Функция Pos определяет номер позиции заданного символа – пробела в исходном тексте T.

С помощью функции Copy выделяется цепочка символов от начала текста до позиции пробела P1. Выделенный фрагмент является первым словом текста. Оператор Writeln(V) выводит первое слово на экран.

Результат работы программы представлен на рисунке 31.

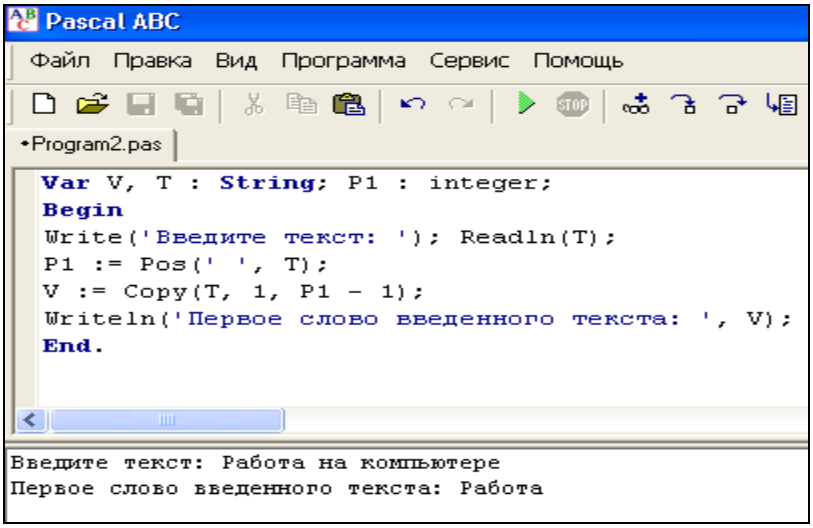



Рисунок 31. Результат работы программы, печатающей первое слово текста

***Пример 2***

*Условие задачи*

Распечатать исходный текст по словам. Слова в тексте разделены символом – пробелом.

Программа на Паскале в этом случае может иметь следующий вид:

```
Const Q = ' ';
Var V, G, T : String; P : Integer;
Begin
Write('Введите текст: '); Readln(T); G := T;
P := Pos(Q, T);
While P <> 0 Do
Begin
  V := Copy(T, 1, P - 1); Writeln(V);
```

```
  Delete(T, 1, P);
  P := Pos(Q, T);
End;
Writeln(T);
End.
```

С помощью оператора цикла While организуем повтор действий: копируем из исходного текста подстроку, ограниченную первым символом и символом – пробелом. Позиция пробела определена в переменной P. Распечатываем подстроку. Удаляем эту строку из исходного текста. Процедура Delete удаляет из строки T P символов, начиная с первого. С помощью функции Pos определяем позицию следующего пробела. Цикл продолжается до тех пор, пока P – позиция пробела не станет равной 0.

Возможно, после последнего пробела останется часть текста, если предположить, что весь текст необязательно заканчивается пробелом. В этом случае эту часть текста необходимо вывести. Для этого предназначен последний оператор вывода Writeln(T).

Результат работы программы представлен на рисунке 32.

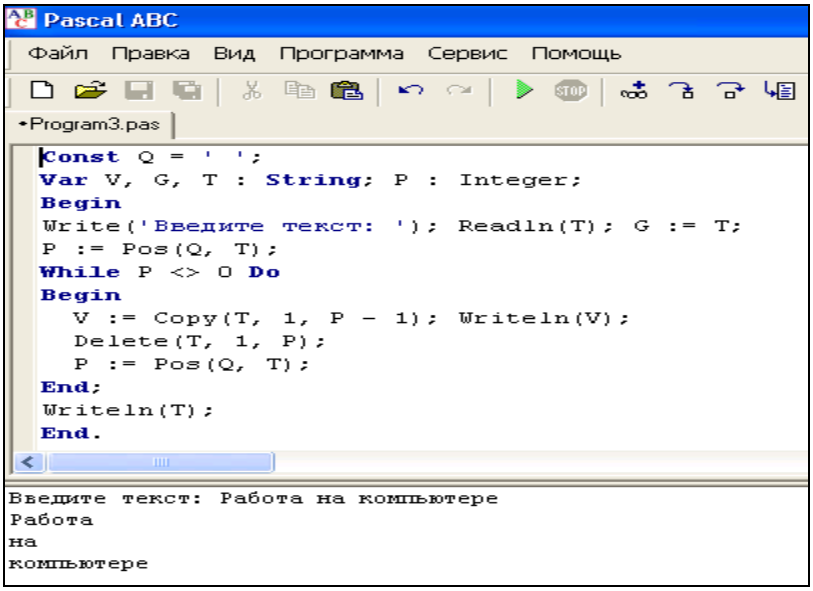



Рисунок 32. Результат работы программы, печатающей исходный текст по словам

***Пример 3***

*Условие задачи*

Выяснить, является ли буква гласной или согласной.

Чтобы выяснить, является буква гласной или согласной, нужно сравнить ее с элементами подготовленного заранее массива гласных или согласных. Массив гласных предпочтительнее – требует меньше памяти.

```
Const Str_1 = 'йуеыаоэяиюё';
Var a : Char;
. . .
If Pos(a, Str_1) <>0 Then Writeln('Гласная')
Else Writeln('Согласная');
```

В разделе описания констант опишем строковую константу. Эта строковая константа содержит в качестве своих символов только гласные. В тексте программы достаточно проверить номер позиции буквы в строке Str_1. Если буква совпадает с каким–либо элементом массива гласных, т.е. позиция вхождения этой буквы в строку не равна 0, то это означает, что буква гласная.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу, определяющую длину введенной строки.

2. Составить программу, определяющую, сколько слов во введенном тексте начинается с буквы К. В тексте слова разделены пробелами.

3. Составить программу, определяющую, является ли введенный текст десятичной записью вещественного числа.

*Вариант 2*

1. Составить программу, определяющую длину введенной строки.

2. Составить программу, определяющую, является ли введенный текст идентификатором.

3. Составить программу, удаляющую часть введенного текста, заключенную в скобки (вместе со скобками).

*Вариант 3*

1. Составить программу, определяющую длину первого слова введенной строки. В тексте слова разделены пробелами.

2. Составить программу, определяющую количество символов * во введенном тексте.

3. Составить программу, определяющую, сколько раз во введенном тексте содержится группа букв abc.

*Вариант 4*

1. Составить программу, определяющую длину последнего слова введенной строки. В тексте слова разделены пробелами.

2. Составить программу, заменяющую точку во введенном тексте восклицательным знаком.

3. Составить программу, выводящую на экран символы, расположенные во введенном тексте внутри скобок (в тексте только одна открывающая скобка и одна закрывающая скобка).

*Вариант 5*

1. Составить программу, заменяющую во введенном тексте двоеточие на точку с запятой.

2. Составить программу, определяющую, содержит ли введенный текст символы, отличные от букв и пробела.

3. Составить программу, определяющую во введенном тексте количество символов, предшествующих двоеточию.

## Лабораторный практикум № 6. Программирование задач обработки записных типов данных

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение основных приемов программирования задач обработки данных типа запись.

***Вводная часть***

*Тип данных «Запись»*

Запись представляет собой совокупность ограниченного числа логически связанных компонент, принадлежащих к разным типам. Компоненты записи называются полями, каждое из которых определяется именем. Поля записи могут относиться к любому типу, допустимому в языке Паскаль, за исключением файлового типа.

Описание записи в языке Паскаль осуществляется с помощью служебного слова record, вслед за которым описываются компоненты записи. Завершается описание записи служебным словом end.

Например, телефонный справочник содержит фамилии и номера телефонов, поэтому отдельную строку в таком справочнике удобно представить в виде следующей записи:

```
type TRec = Record
    FIO: String[20];
```

```
    TEL: String[11]
  end;
var Rec: TRec;
```

Описание записей возможно и без использования имени типа, например:

```
var Rec: Record
      FIO: String[20];
      TEL: String[11]
    end;
```

*Обращение к данным типа «Запись». Оператор присоединения With*

Обращение к записи в целом допускается только в операторах присваивания, где слева и справа от знака присваивания используются имена записей одинакового типа. Во всех остальных случаях оперируют отдельными полями записей. Чтобы обратиться к отдельной компоненте записи, необходимо задать имя записи и через точку указать имя нужного поля, например:

Rec.FIO или Rec.TEL

Например, Rec.FIO:= ‘Иванов П.П.’ или Rec.TEL:=’89161234567’.

Такое имя называется составным.

Обращение к компонентам записей можно упростить, если воспользоваться оператором присоединения With. Он позволяет заменить составные имена, характеризующие каждое поле, просто на имена полей, а имя записи определить в операторе присоединения:

with Rec do <оператор>;

Здесь Rec - имя записи, оператор - простой или составной оператор. Оператор представляет собой область действия оператора присоединения, в пределах которой можно не использовать составные имена. Например, для нашего случая:

```
with Rec do begin
      FIO:='Иванов П.П.';
      TEL:='89161234567';
end;
```

Такая алгоритмическая конструкция полностью идентична следующей:

```
Rec.FIO:='Иванов П.П.';
Rec.TEL:='89161234567';
```

Для работы с информацией, представленной в табличной форме, часто используют массив записей:

```
Var Tel_book: array [1..50] of TRec;
```

***Практическая часть***

***Пример 1***

*Условие задачи*

Организовать массив записей, содержащий информацию о сотрудниках отдела: Табельный номер; ФИО; Стаж работы; Зарплата. Выдать на экран данные о сотрудниках, у которых стаж работы составляет не менее 10 лет.

| Табельный номер | ФИО | Стаж работы | Зарплата |
|---|---|---|---|
| 1 | Иванов А.П | 12 | 18000 |
| 2 | Петров С.Л. | 15 | 21000 |
| 3 | Сидоров К.М. | 8 | 12000 |
| 4 | Васин П.Д. | 11 | 15000 |

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Spisok_otd;
```

```
Const nn=20;
Type                                              {Объявление типа запись}
  Sotr=Record
     Tab_Nom:integer;
     Fio: string[15];
     Stag_rab, Zarp: integer;
  end;
Var   Spisok: Array [1..nn] of Sotr;              {Описание массива записей}
      i,n:integer;
BEGIN
  Write('Число сотрудников?  '); Readln(n);
  For i:=1 to n do                                {формирование массива  данных о
                                                    сотрудниках}
    Begin
       Write('Номер сотрудника ');  Readln(Spisok[i].Tab_Nom);
       Write('Ф.И.О. ');  Readln(Spisok[i].Fio);
       Write('Стаж ');  Readln(Spisok[i].Stag_rab);
       Write('Зарплата ');  Readln(Spisok[i].Zarp);
    End;
  writeln('Tab_Nom  ','      Fio     ',' Stag_rab  ','   Zarp');      {формирование шапки таблицы}
  For i:=1 to n do       {вывод информации о сотрудниках,
                                                    стаж работы которых превышает 10 лет}
    With  Spisok[i] do     {оператор with используется для
                                                      упрощения работы с полями записи }
       If stag_rab>=10 then
       begin
          writeln(Tab_Nom:7, Fio:15, Stag_rab:10, Zarp:10);
       END;
END.
```

Результат работы программы представлен на рисунке 33.

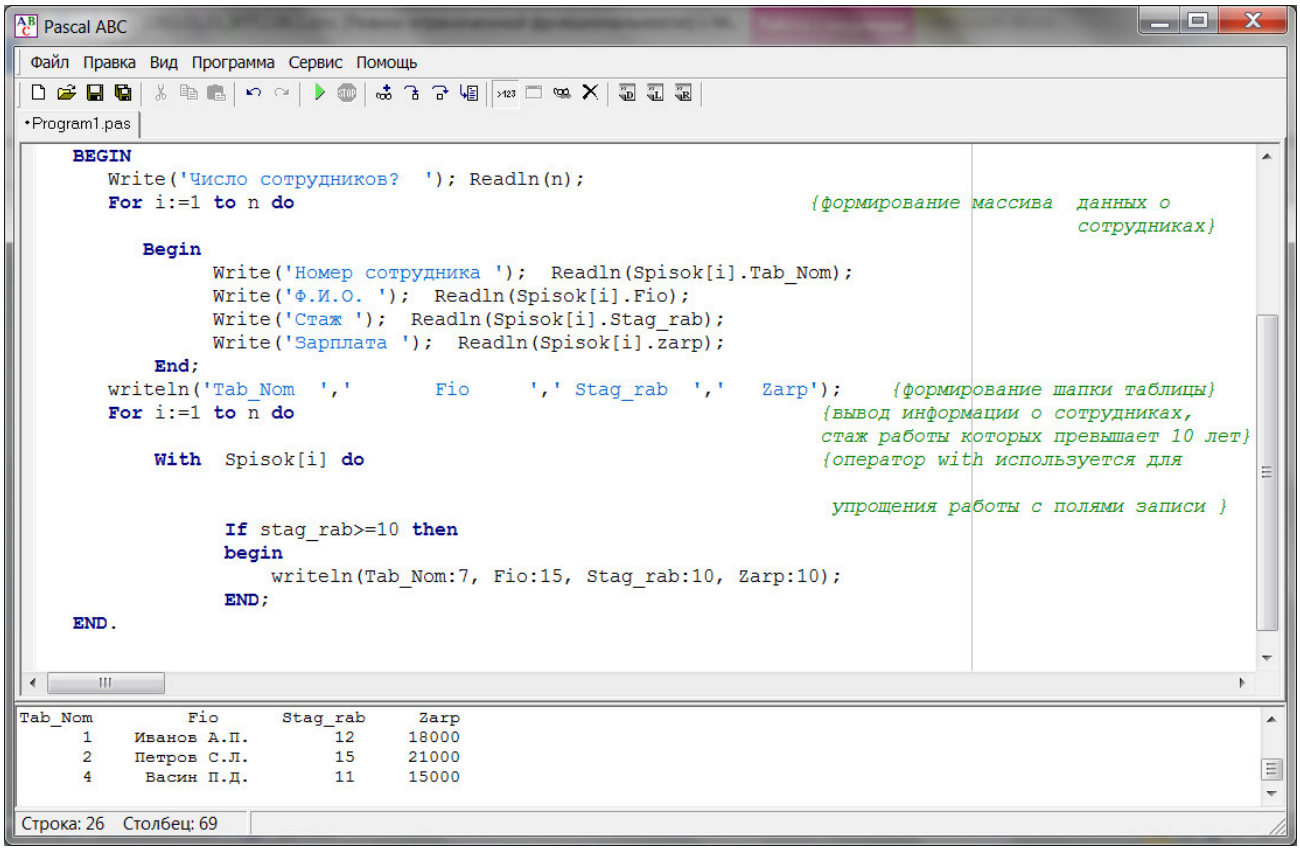



Рисунок 33. Результат работы программы, представленной в примере 1

***Пример 2***

*Условие задачи*

Организовать массив записей, содержащий информацию о друзьях: Фамилия; Имя; Телефон; Адрес. По введенной с клавиатуры фамилии и имени организовать поиск и вывод на экран номера телефона и адреса друга.

| Фамилия | Имя | Телефон | Адрес |
|---|---|---|---|
| Иванов | Алексей | 12-56-67 | Озерная, д. 3, кв. 5 |
| Петров | Сергей | 15-55-33 | Озерная, д. 3, кв. 9 |
| Сидоров | Константин | 83-43-23 | Садовая, д. 6, кв. 2 |
| Васин | Павел | 11-33-44 | Лесная, д. 7, кв. 11 |

Для решения задачи заведем переменную Flag логического типа. Присвоим этой переменной значение False. Будем просматривать записи телефонной книги для поиска и вывода на экран номера телефона и адреса друга, имя которого ввели по запросу. Если искомая информация в телефонной книге обнаружена, переменной Flag присваивается значение True и на экран выводятся найденный номер телефона и адрес. Если искомой информации в телефонной книге обнаружено не было, то после просмотра всех записей книги значение переменной Flag продолжает оставаться False. Проверим значение Flag после окончания цикла. Если Flag = False, то выведем на экран информацию о том, что искомой информации в телефонной книге обнаружено не было.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Tel_Book;
  Const nn=20;
  Type TRec = Record
```

```
        Fam: String[10];
        Im:  String[10];
        Tel: String[8];
        Adres: String[20];
    end;
Var T_Book: Array [1..nn] of TRec;
   Famil, Name: String[10];
   i,n: Integer;
   Flag:  Boolean;                    {Эта переменная понадобится для проверки наличия
                                искомой информации в телефонной книге}
BEGIN
  Write('Введите число записей в тел. книге');    Readln(n);
  For i:=1 to n do                {формирование массива  записей телефонной книги}
    Begin
      Write('Фамилия?'); Readln(T_Book[i].Fam);
      Write('Имя?'); Readln(T_Book[i].Im);
      Write('Телефон? '); Readln(T_Book[i].Tel);
      Write('Адрес? '); Readln(T_Book[i].Adres);
    End;
                                                {запрос информации для поиска}
  Write('Введите фамилию для поиска');
  Readln(Famil);
  Write('Введите имя для поиска');
  Readln(Name);
                                          {поиск и вывод номера телефона и адреса}
  Flag:= False;
  For i:=1 to n do
    With  T_Book[i] do
      If (Fam = Famil) And (Im = Name) then
      begin
        Flag:= True;
        writeln('Телефон:   ', Tel);
        writeln('Адрес:    ',  Adres)
      end;
  If  Flag = False  then  Writeln ('В телефонной книге искомой информации не обнаружено')
end.
```

Результат работы программы представлен на рисунке 34.

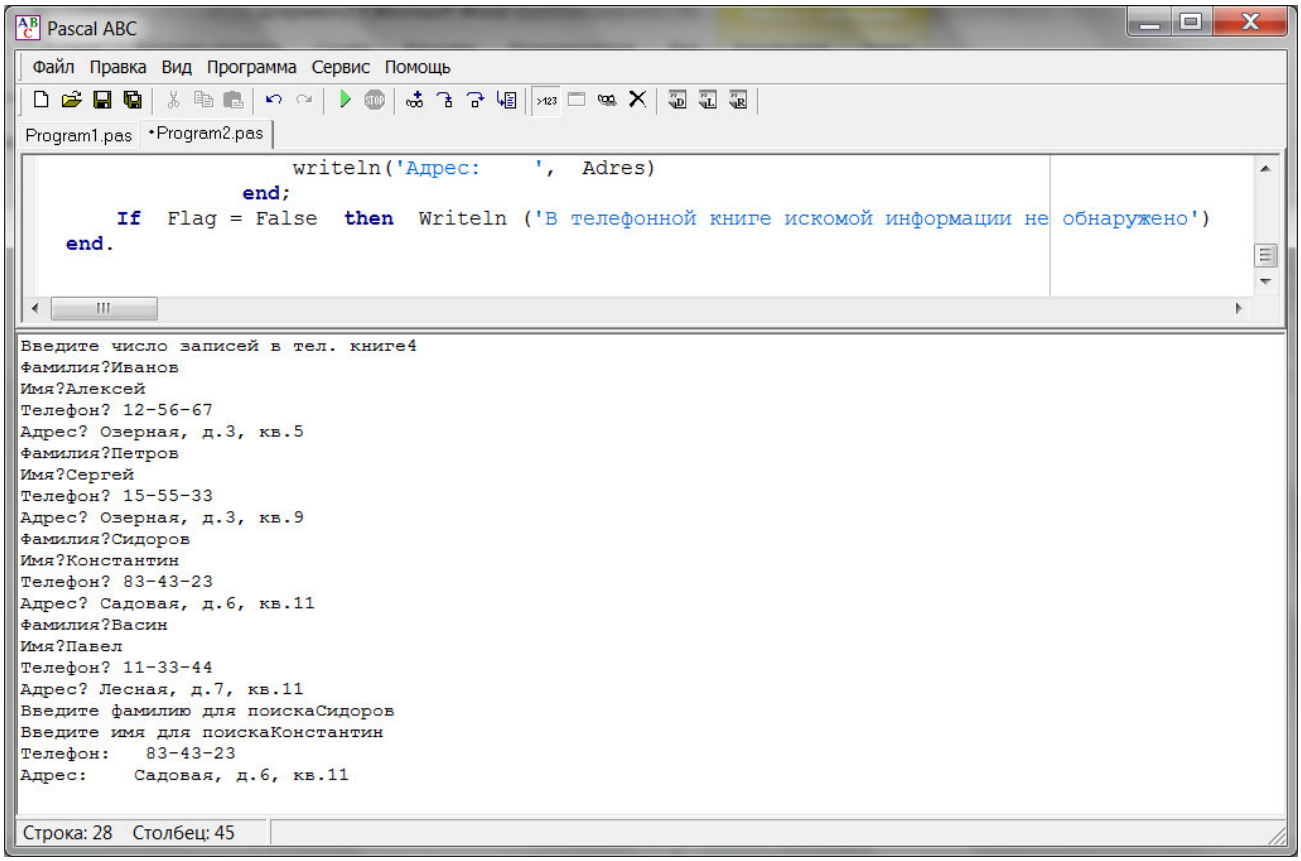



Рисунок 34. Результат работы программы, представленной в примере 2

***Самостоятельная работа***

*Вариант 1*

1. Организовать массив записей, содержащий информацию о семьях, имеющих детей. Каждая запись должна содержать следующую информацию: ФИО (глава семьи); улица; № дома; № квартиры; количество детей. Вывести на экран список семей, которые должны получать пособие для многодетных (более двух детей), с указанием ФИО главы семьи и адреса.

2. Организовать массив записей, содержащий информацию об обучающихся: ФИО, номер группы, оценка по информатике, оценка по математике, оценка по физике, средний балл (расчетный параметр). Заполнить поле «Средний балл» для каждого обучающегося. Подсчитать количество обучающихся, имеющих средний балл, ниже трех. Вывести на экран полную информацию об обучающемся, ФИО которого вводится с клавиатуры.

*Вариант 2*

1. Организовать массив записей, содержащий информацию о товарах, хранящихся на складе. Каждая запись должна содержать следующую информацию: индекс товара, наименование товара, цена товара, количество товара. Вывести на экран индексы и наименования товаров, цена которых превышает 10000 рублей.

2. Организовать массив записей, содержащий информацию о лекарствах в аптеке: название лекарства, страна-производитель, цена, количество, стоимость (расчетный параметр). Подсчитать общую стоимость лекарств в аптеке. Найти количество лекарств, произведенных в России. Вывести справку о наличии конкретного лекарства в аптеке. Наименование лекарства вводится с клавиатуры.

*Вариант 3*

1. Организовать массив записей, содержащий информацию об автовладельцах: ФИО владельца; номер машины; марка машины; год изготовления. Вывести на экран ФИО владельца, номер и марку тех машин, которые были изготовлены до 2010 года.

2. Организовать массив записей, содержащий информацию о сотрудниках двух отделов: ФИО сотрудника, номер отдела, зарплата, премия, начислено (расчетный параметр). Заполнить поле

«Начислено» для каждого сотрудника. Вычислить общую сумму, начисленную каждому отделу. Вывести информацию о сотруднике, ФИО которого вводится с клавиатуры.

*Вариант 4*

1. Организовать массив записей, содержащий информацию об обучающихся, сдававших экзамен: ФИО обучающегося, номер студенческого билета, номер группы, оценка за экзамен. Вывести на экран список обучающихся, получивших на экзамене отличную оценку, с указанием ФИО и номера группы.

2. Организовать массив записей, содержащий информацию о товарах на складе: артикул товара, наименование товара, цена единицы товара, количество товара, стоимость товара (расчетный параметр). Заполнить поле «Стоимость товара» для каждого товара. Найти общую стоимость всех платьев (различные артикулы). Вывести полную информацию о товаре, наименование и артикул которого вводится с клавиатуры.

*Вариант 5*

1. Организовать массив записей, содержащий информацию об абитуриентах: номер личного дела; ФИО; факультет; нуждается ли в общежитии. Вывести на экран ФИО абитуриентов, нуждающихся в общежитии, и название факультета, на который они поступают.

2. Организовать массив записей, содержащий информацию о пассажирах: ФИО пассажира, номер рейса, место назначения, количество багажа, вес багажа. Вычислить суммарный вес багажа рейса 1235. Найти количество пассажиров, улетающих во Владивосток. Вывести полную информацию о пассажире, ФИО которого вводится с клавиатуры.

## Лабораторный практикум № 7. Программирование задач с использованием подпрограмм-функций

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 3.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение основных приемов программирования задач с использованием подпрограмм-функций.

***Вводная часть***

*Понятие подпрограммы*

При разработке программ иногда требуется одни и те же последовательности действий выполнять на различных этапах обработки информации. В таких задачах в различных местах встречаются фрагменты, одинаковые по выполняемым действиями, различающиеся только в значениях исходных данных. Повторяющаяся группа операторов оформляется в виде самостоятельной программной единицы – подпрограммы. Подпрограмма – это самостоятельная часть программы, реализующая определенный алгоритм и допускающая обращение к ней из различных частей основной программы. Так, например, может возникнуть необходимость один и тот же цикл использовать в нескольких местах программы.

В языке Pascal подпрограммы реализуются в виде процедур и функций, которые вводятся в программу с помощью своего описания.

*Структура подпрограммы. Локальные и глобальные параметры*

Функцию или процедуру можно сравнить с мини-программой, именно поэтому их называют иногда одним общим именем - "подпрограмма". Подпрограмма оформляется подобно программе: в начале записывается заголовок подпрограммы, затем следует декларативная часть и после процедурная. В декларативной части описываются все данные, область действия которых ограничена телом данной подпрограммы. Эти данные называются локальными. Данные, объявленные в основной (главной) программе, называются глобальными, и они могут использоваться в любой подпрограмме, входящей в основную программу. В процедурной части

описывается тело подпрограммы, реализующее алгоритм решения и которое заключается в операторные скобки BEGIN, END.

Подпрограмма помещается сразу же после объявления всех переменных в основной (главной) программе. Заголовок подпрограммы для подпрограмм-функций начинается с ключевого слова FUNCTION, для подпрограмм-процедур с ключевого слова PROCEDURE. Эти ключевые слова играют роль признаков подпрограммы, которые распознает компилятор. Так как подпрограмма исполняется не сразу и, возможно, не один раз, то компилятор, встретив ее тело, должен его пропустить.

*Формат описания подпрограммы-функции*

Формат описания подпрограммы-функции следующий:

```
function имя функции (формальные параметры) : тип результата;
        раздел описаний функции
begin
        исполняемая часть функции
end;
```

Более подробное описание:

```
Function имя функции (формальные параметры) : тип результата функции;
   LABEL …;
   CONST …;
   TYPE …;
   VAR …;                {объявление локальных переменных}
Begin
   { тело функции }
End;
```

Самым простым и наглядным примером использования функций являются стандартные функции, например функция LENGTH(St). Эта функция может применяться в программе всякий раз, когда необходимо вычислить длину строки, в данном случае строковой переменной St. Все стандартные функции входят в состав компилятора, то есть они описаны в теле самого компилятора. LENGTH - это идентификатор функции, St - аргумент функции. Заголовок этой функции может быть следующий:

```
FUNCTION Lenght(St : string) : byte;
```

Тип результата, возвращаемого этой функцией, BYTE.

Вызов функции в программе может осуществляться выражением:

N:= Lenght(Str); где переменная N описана как BYTE, а Str – как string.

*Структура программы, содержащей подпрограмму-функцию*

Структуру программы, содержащей подпрограмму-функцию, можно представить следующим образом:

```
PROGRAM Main;
CONST…;                          { декларативная часть главной программы}
VAR …   ;
                                 { процедурная часть главной программы }
FUNCTION Sample( . . . ) : . . .;  { заголовок функции }
VAR … ;                          { декларативная часть подпрограммы }
BEGIN { Sample }                  { процедурная часть подпрограммы }
…
Sample:= … ;
END; { Sample }
BEGIN                             { главная программа }
…
< любое количество вызовов функции Sample >
…
```

END. { конец главной программы }

*Формальные и фактические параметры*

При описании функции в скобках указываются формальные параметры, при вызове функции – фактические. При вызове функции фактические параметры как бы подставляются вместо формальных, стоящих на тех же местах в заголовке. Таким образом, происходит передача входных параметров, затем выполняются операторы исполняемой части подпрограммы, после чего происходит возврат в вызывающий блок. Передача выходных параметров происходит непосредственно во время работы исполняемой части.

Например:

1. ЗАГОЛОВОК ФУНКЦИИ

FUNCTION F ( N: REAL): REAL; {N – формальный параметр}

2. ВЫЗОВ ФУНКЦИИ

PER: = F (K); {K – фактический параметр}

Для передачи в вызывающий блок выходного значения функции в исполняемой (процедурной) части функции перед возвратом в вызывающий блок необходимо поместить следующую команду:

имя функции := результат;

При вызове процедур и функций необходимо соблюдать следующие правила:

1) количество фактических параметров должно совпадать с количеством формальных;

2) соответствующие фактические и формальные параметры должны совпадать по порядку следования и по типу.

***Практическая часть***

***Пример 1***

*Условие задачи*

Составить программу вычисления факториалов Fn=n!, Fm=m!, Fnm=(n-m)!.

Вычисление факториала оформить в виде подпрограммы-функции с параметрами.

Факториал n! представляет собой произведение n чисел натурального ряда: 1*2*3*...*n.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Example1;
   Var
fn,fm,fnm: integer;
n,m: integer;
  (* функция Fakt *)
  Function Fact (k: integer): integer;      {начало описания функции}
Var
P, i:  integer;                    {раздел описания локальных переменных}
  Begin                                {начало операторной части функции}
P: = 1;
For i: = 1 to  k  do
P: = P*i;
Fact: = P;
  End;                                 {конец описания функции}
  (* основная программа *)
Begin
Write ('введите значения n,m: ');
Read (n,m);                    {ввод данных с клавиатуры}
Fn: = Fact(n);                   {обращение к функции}
Fm: = Fact(m);                   {обращение к функции}
Fnm: = Fact(n-m);                {обращение к функции}
```

```
Writeln ('Fn=',Fn:5);              {вывод результата}
Writeln ('Fm=',Fm:5);              {вывод результата}
Writeln ('Fnm=',Fnm:5);            {вывод результата}
End.                                    {конец программы}
```

Результат работы программы представлен на рисунке 35.

***Пример 2***

*Условие задачи*

Составить функцию Sum, вычисляющую сумму положительных элементов одномерного массива. Использовать функцию Sum для подсчета суммы положительных элементов в двух одномерных массивах.

Для решения задачи составим функцию, в которой в цикле будем "пробегать" массив, сравнивая его элементы с 0, и суммировать, если они больше нуля.

Программа на Паскале в этом случае может иметь следующий вид:

```
program Example2;
    type
      Mass = array[1..100] of integer;
    Var A, B : Mass;
      n, i, SA, SB, SO : integer;
    Function Sum (m : Mass) : integer;
```

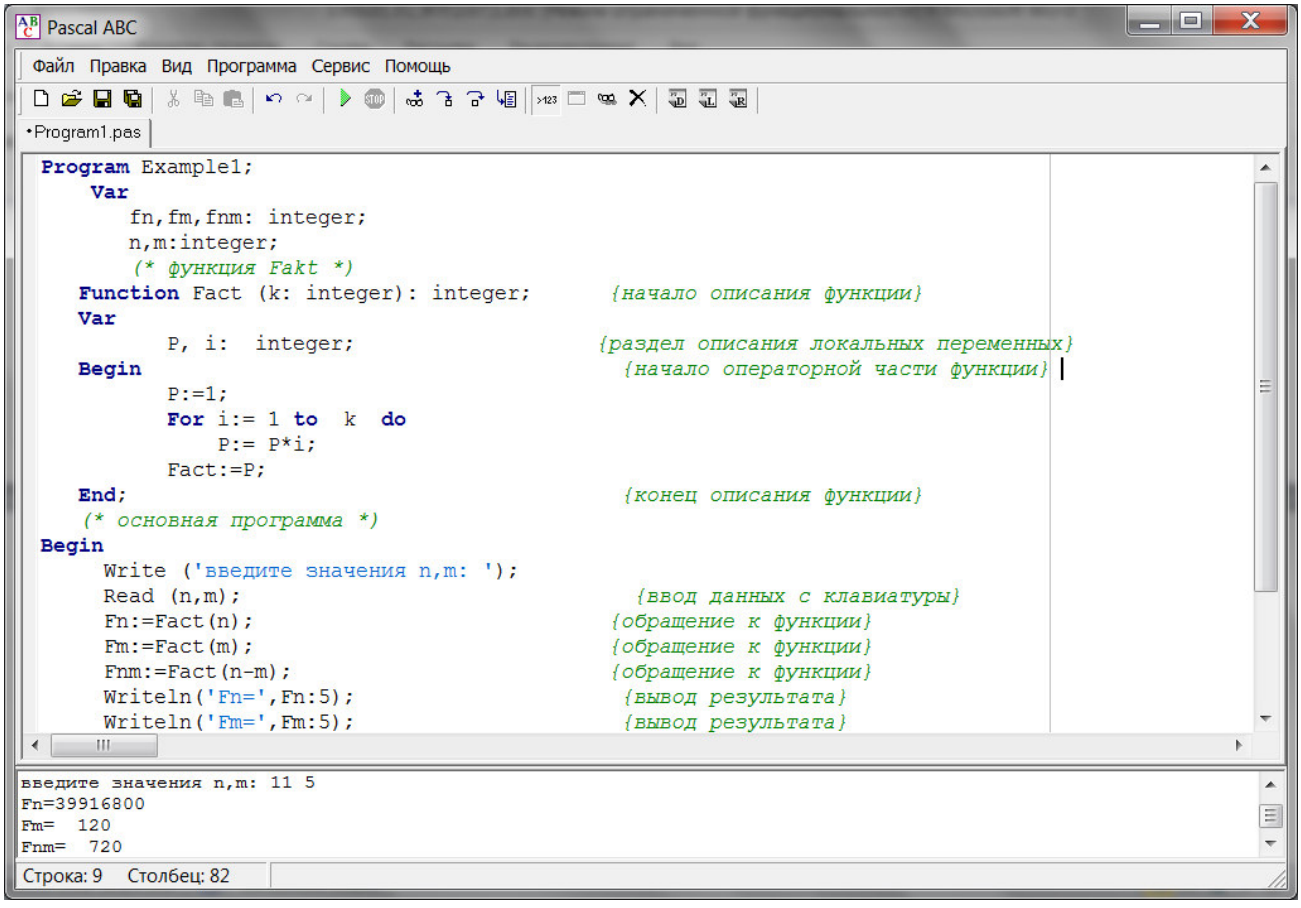



Рисунок 35. Результат работы программы, вычисляющей факториал числа

```
    Var S : integer;
  Begin
    S:=0;
```

```
        For i:= 1 to n do if m[i]>0 then s:= s+m[i];
        Sum:=s;
    End;
begin
    Write('Введите размерность массивов A и B  '); Readln (n);
    Writeln('Введите ',n,' элементов массива A через пробел');
    For i:= 1 to n do read(A[i]);
    Writeln('Введите ',n,' элементов массива B через пробел');
    For i:= 1 to n do Read (B[i]);
    SA:=  Sum(A);
    SB:= Sum (B);
    SO:= SA +SB;
    Writeln('Сумма положительных элементов массива A = ',SA );
    Writeln('Сумма положительных элементов массива B = ',SB );
    Writeln('Сумма положительных элементов двух массивов = ', SO);
end.
```

Результат работы программы представлен на рисунке 36.

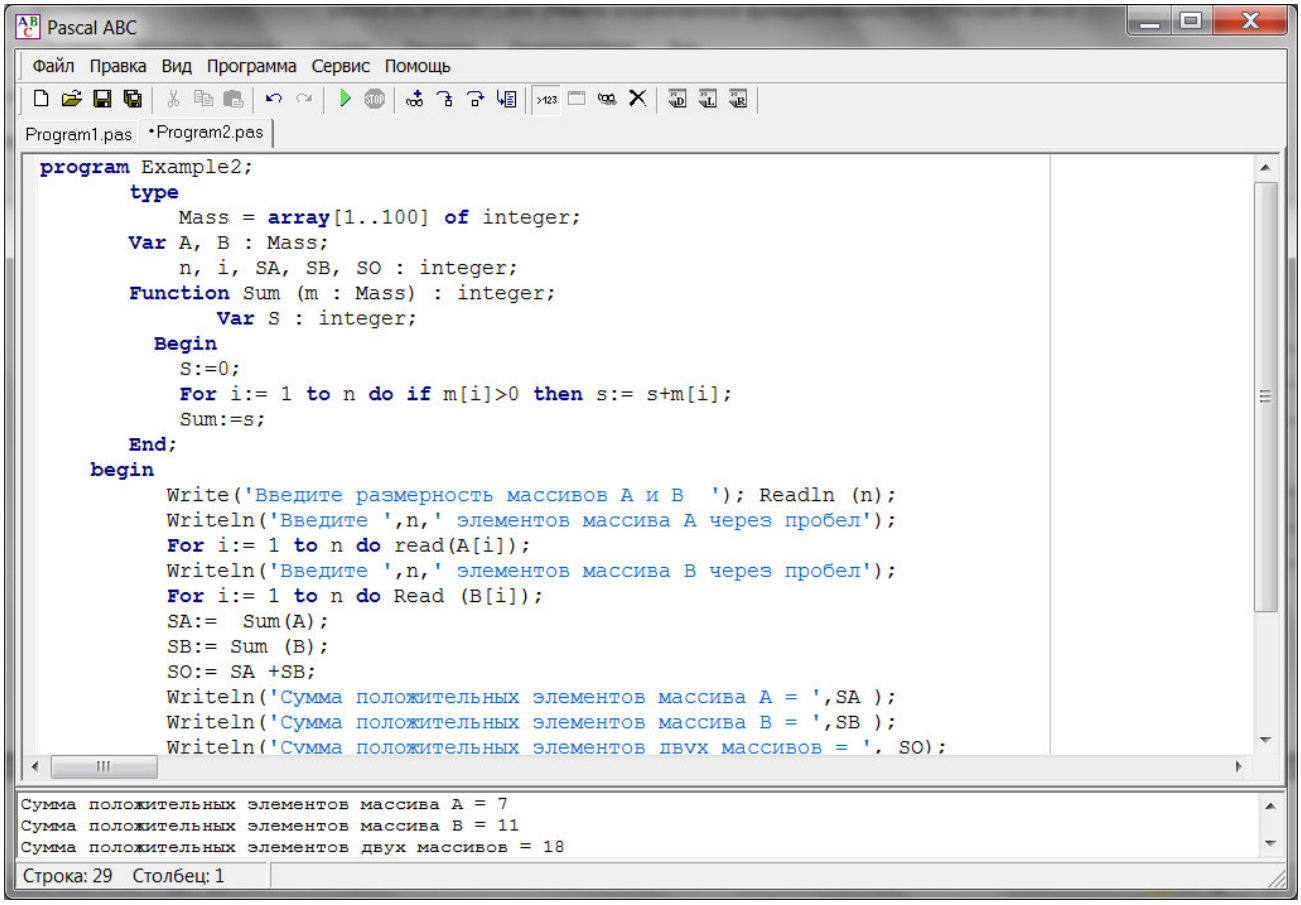



Рисунок 36. Результат работы программы, представленной в примере 2

***Самостоятельная работа***

*Вариант 1*

1. Используя подпрограмму-функцию, вычисляющую десятичный логарифм lg(x), составьте программу вычисления У.

$$Y = \frac{\lg a - \lg b}{2 \bullet \lg (a + b)}.$$

Для подпрограммы-функции использовать соотношение lg x = 0.4343 * ln x.

2. Используя подпрограмму-функцию, определяющую количество элементов, равных нулю, в одномерном массиве, вычислить общее количество нулевых элементов в трех разных массивах A, B и C одной размерности.

*Вариант 2*

1. Используя подпрограмму-функцию, вычисляющую сумму чисел натурального ряда, составьте программу вычисления У.

$$Y = \frac{\sum_{i=1}^{n} i - \sum_{j=1}^{m} j}{\sum_{k=1}^{m+n} k}.$$

Пояснение : $\sum_{a=1}^{b} a = 1 + 2 + 3 + ... + b$.

2. Составить подпрограмму- функцию, позволяющую определить количество цифр в заданной строке. Использовать эту функцию для определения общего количества цифр в трех разных строках.

*Вариант 3*

1. Найти наибольшее из четырех чисел, используя подпрограмму-функцию нахождения наибольшего из двух чисел.

2. Составить подпрограмму-функцию, определяющую сумму отрицательных элементов в двумерном массиве. Использовать эту функцию для подсчета общей суммы отрицательных элементов в двух разных массивах одинаковой размерности.

*Вариант 4*

1. Используя подпрограмму-функцию, вычисляющую степень числа, составьте программу вычисления Z.

$$Z = \frac{X^n + Y^m}{(X - Y)^{m+n}},$$

где X, Y, m и n – натуральные числа.

2. Составить подпрограмму-функцию, позволяющую определить количество вхождений заданного символа в исходную строку. С помощью этой функции определить, каких символов: a, b или c в заданной строке содержится больше.

*Вариант 5*

1. Используя подпрограмму-функцию, вычисляющую произведение чисел натурального ряда, составьте программу вычисления У.

$$Y = \prod_{i=2}^{n} i + \prod_{j=3}^{m} j + \prod_{k=1}^{n+m} k.$$

Пояснение : $\prod_{a=1}^{b} a = 1 \bullet 2 \bullet 3 \bullet ... \bullet b$.

2. Составить подпрограмму-функцию, позволяющую определить максимальный элемент одномерного массива. С помощью этой функции определить максимальные элементы в трех разных массивах.

**Лабораторный практикум № 8. Программирование задач с использованием процедур**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 3.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение основных приемов программирования задач с использованием подпрограмм-процедур.

***Вводная часть***

*Понятие подпрограммы-процедуры*

Подпрограмма – повторяющаяся группа операторов, которая оформляется в виде самостоятельной программной единицы. Подпрограмма реализует определенный алгоритм и допускает обращение к ней из различных частей основной программы. В языке Паскаль подпрограммы реализуются в виде процедур и функций, которые вводятся в программу с помощью своего описания.

Процедуры по своему внешнему виду напоминают функции, но если при обращении к функции можно получить только один результат – значение данной функции, то процедура может вырабатывать на выходе несколько значений.

*Стандартные процедуры*

Процедуры могут использоваться в программах для выполнения ряда стандартных действий. Такие процедуры, подобно стандартным функциям, являются частью языка Паскаль.
К стандартным процедурам относятся ClrScr, TextColor и TextBackground и многие другие. Процедурами являются и операторы ввода и вывода read, readln, write и writeln. Наряду с этими стандартными процедурами в языке Паскаль широко используются также следующие:

• Delay (i) – осуществляет задержку выполнения программы на i миллисекунд (тысячных долей секунды);

• Exit – эта процедура не имеет параметров. Она осуществляет выход из процедуры или функции в основную часть программы;

• GotoXY (a, b) – переводит курсор в точку экрана с координатами a, b;

• Halt – эта процедура не имеет параметров. Она завершает выполнение программы и передает управление операционной системе;

• Str(x,s) – производит преобразование числовой величины x в строковую s;

• Val(s,x,n) – производит преобразование строки s, изображающей число, в числовую величину x. В процедуре также должен присутствовать параметр n, относящийся к целочисленному типу. Если преобразование было выполнено успешно, то значение n будет равно нулю. Если преобразование не может быть произведено, то в переменную n записывается номер символа, который явился причиной ошибки при преобразовании.

*Формат описания подпрограммы-функции*

Возможности языка Паскаль не ограничиваются применением только стандартных процедур. Программист может создавать и свои собственные процедуры. Но для того, чтобы такую процедуру можно было использовать в программе, ее, подобно вновь создаваемым функциям, следует предварительно описать. Описание процедуры, так же как и описание функции, должно содержаться в программе в разделе описаний после описания констант и переменных. Структура описания процедуры сходна со структурой основной программы, поэтому создаваемую программистом процедуру часто называют подпрограммой-процедурой. Описание включает в себя заголовок процедуры, раздел описаний и раздел операторов. К процедуре можно обращаться из основной части программы, из другой процедуры или из функции. Такое обращение называют вызовом процедуры.

*Формат описания подпрограммы-процедуры*

Формат описания подпрограммы-процедуры следующий:

```
procedure имя процедуры (формальные параметры);
    раздел описаний процедуры
begin
```

исполняемая часть процедуры

end;

Вызов процедуры производится оператором, имеющим следующий формат:

имя процедуры(список фактических параметров);

Разберем более подробно составные части данного описания.

Общий вид заголовка процедуры следующий:

procedure имя_процедуры(формальные параметры процедуры);

где procedure – служебное слово, имя процедуры дается по тем же правилам, что и имя переменной в Паскале, параметры перечисляются в скобках через запятую с указанием их типа. Эти параметры являются формальными. При вызове же процедуры в обращении к ней указываются ее фактические параметры. Тип каждого фактического параметра должен быть таким же, как тип соответствующего ему формального параметра. Количество формальных параметров, имеющихся в описании процедуры, и фактических параметров, используемых при обращении к ней, должно совпадать. При этом первому по счету формальному параметру в описании ставится в соответствие первый по счету фактический параметр в обращении, второму формальному – второй фактический и т.д.

*Параметры-значения и параметры-переменные*

Все формальные параметры делятся на два вида. Если перед именем параметра в заголовке процедуры стоит служебное слово var, то это – параметр-переменная. Если служебное слово var перед именем переменной в заголовке отсутствует, то данный параметр является параметром-значением. При обращении к процедуре формальному параметру-значению присваивается значение соответствующего ему фактического параметра, причем в качестве такого значения может выступать константа, переменная или выражение. Во время работы процедуры параметр-значение не может изменяться даже в том случае, если он является переменной. При обращении же к процедуре, в которой имеются формальные параметры-переменные, соответствующие им фактические параметры могут быть только переменными (не константами и не выражениями). Вызываемая процедура получает доступ к ячейкам памяти, в которых хранятся эти фактические параметры, и может изменять значения этих параметров в ходе своей работы.

Пример заголовка процедуры:

procedure vspomog(a,b:integer; var c,d:real);

где vspomog – имя процедуры,

a, b, c, d – имена формальных параметров, причем a и b являются параметрами-значениями, a c и d – параметрами-переменными.

В разделе описаний процедуры константы и переменные описываются по тем же правилам, что и в основной программе. При этом следует иметь в виду, что эти переменные могут использоваться только внутри данной процедуры. Такие переменные называются локальными.
В процедуре могут применяться и переменные, которые описаны в основной части программы. Такие переменные называются глобальными.

Раздел операторов процедуры принципиально не отличается от такого же раздела в основной программе, он может содержать обращения к другим функциям и процедурам, но после служебного слова end ставится не точка, а точка с запятой, так как конец описания процедуры – это не конец программы.

*Структура программы, содержащей подпрограмму-процедуру*

Структуру программы, содержащей подпрограмму-процедуру, можно представить следующим образом:

```
PROGRAM Main;
CONST…;                          { декларативная часть главной программы}
VAR …  ;
                                 { процедурная часть главной программы }
PROCEDURE  Sample ( . . . );     { заголовок процедуры }
VAR … ;                          { декларативная часть подпрограммы }
BEGIN { Sample }                 { процедурная часть подпрограммы }
```

```
…
…
END; { Sample }
BEGIN                                   { главная программа }
…
Sample (…);          < любое количество вызовов процедуры  Sample >
…
END.                                    { конец главной программы }
```

*Формальные и фактические параметры*

При описании процедуры в скобках указываются формальные параметры, при вызове процедуры – фактические. При вызове процедуры фактические параметры как бы подставляются вместо формальных, стоящих на тех же местах в заголовке. Таким образом, происходит передача входных параметров, затем выполняются операторы исполняемой части подпрограммы, после чего происходит возврат в вызывающий блок. Передача выходных параметров происходит непосредственно во время работы исполняемой части.

Например:

1. ЗАГОЛОВОК ПРОЦЕДУРЫ

```
PROCEDURE PR( N, M: REAL;VAR A, B: REAL);       {N, M, A, B – формальные параметры}
```

2. ВЫЗОВ ФУНКЦИИ

```
PR(K,L,X,Y);                            {K,L,X,Y – фактические параметры}
```

При вызове процедур и функций необходимо соблюдать следующие правила:

1) количество фактических параметров должно совпадать с количеством формальных;

2) соответствующие фактические и формальные параметры должны совпадать по порядку следования и по типу.

***Практическая часть***

***Пример 1***

*Условие задачи*

Найти наибольшее из четырех чисел, используя подпрограмму-процедуру нахождения наибольшего из двух чисел.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Example1;
    var a, b, c, d, m, n, maximum: integer;
    procedure MaxNumber(x,y: integer; var max: integer);    {описание процедуры}
    begin
        if x>y then max:=x  else  max:=y;                   {определение большего из двух чисел}
    end;
begin
     write('Введите  a,b,c,d ');
     readln(a,b,c,d);                {ввод четырех чисел}
     MaxNumber(a,b,m);       {вызов процедуры для определения большего из чисел  a и b }
     MaxNumber(c,d,n);       {вызов процедуры для определения большего из чисел  c и d }
     MaxNumber(m,n,maximum); {вызов процедуры для определения большего из четырех
                                                                                чисел}
    writeln('maximum = ',maximum);                  {вывод результата}
end.
```

Результат работы программы представлен на рисунке 37.

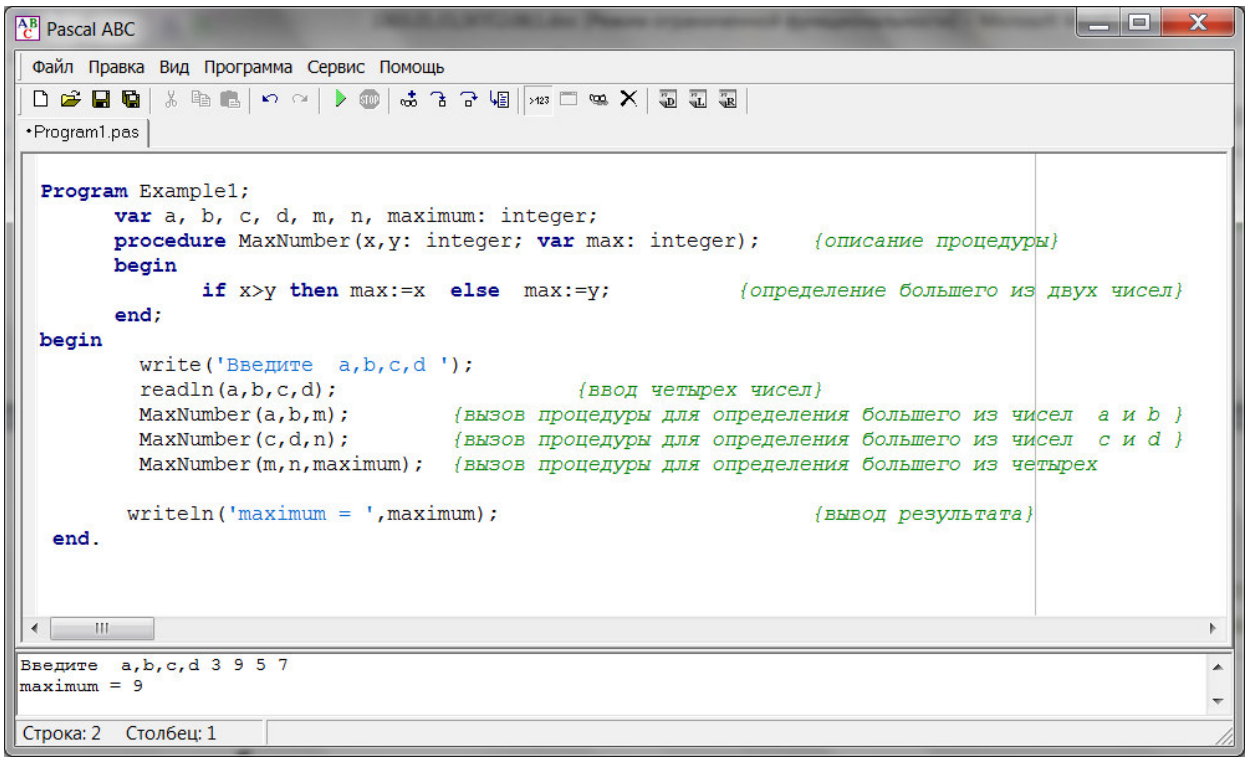



Рисунок 37. Результат работы программы, представленной в примере 1

***Пример 2***

*Условие задачи*

Составить процедуру, подсчитывающую сумму и количество положительных элементов одномерного массива. Использовать эту процедуру для подсчета общей суммы и общего количества положительных элементов в двух одномерных массивах.

Для решения задачи составим процедуру, в которой в цикле будем "пробегать" массив, сравнивая его элементы с нулем, и подсчитывать сумму и количество тех элементов, значения которых больше нуля.

Программа на Паскале в этом случае может иметь следующий вид:

```
program Example2;
    type                                        {объявление массивового типа}
      Mass = array[1..100] of integer;
    Var A, B : Mass;
       n, i, SummaA, KolvoA, SummaB, KolvoB  : integer;
       Summa,  Kolvo :  integer;
    Procedure Sum_Kol(m : Mass; Var S,K : integer );   {описание процедуры,
подсчитывающей сумму S и количество K положительных элементов массива}
       Begin
         S:=0; K:=0;                        {обнуляем значения переменных S и K}
         For  i:=1  to  n  do  if  m[i]>0  then                    {последовательно  сравниваем  элементы
                                                  массива с нулем}
           begin S:=S+m[i];                 {если элемент положителен, суммируем его}
                 K:=K+1;                    {и увеличиваем переменную K на единицу}
           end;
       End;
 begin
      Write('Введите размерность массивов A и B  '); Readln (n);
```

```
Writeln('Введите ',n,' элементов массива A через пробел');
For i:=1 to n do read(A[i]);
Writeln('Введите ',n,' элементов массива B через пробел');
For i:=1 to n do Read (B[i]);
Sum_Kol(A, SummaA, KolvoA);   {вызов процедуры для подсчета суммы и
                               количества положительных элементов массива A}
Sum_Kol(B, SummaB, KolvoB);   {вызов процедуры для подсчета суммы и
                               количества положительных элементов массива B}
 Summa:= SummaA + SummaB;
Kolvo:=KolvoA + KolvoB;
Writeln(' Сумма положительных элементов двух массивов = ', Summa);
Writeln(' Количество положительных элементов двух массивов = ', Kolvo);
end.
```

Результат работы программы представлен на рисунке 38.

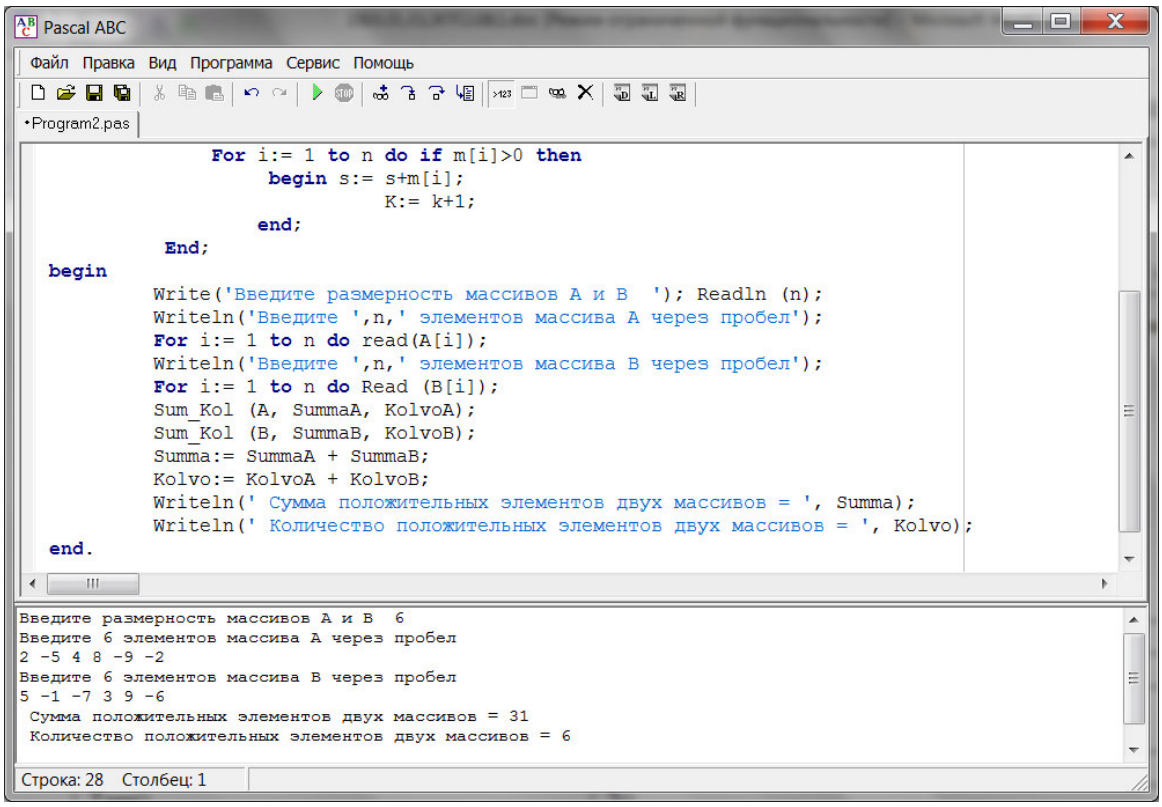



Рисунок 38. Результат работы программы, представленной в примере 2

***Самостоятельная работа***

*Вариант 1*

1. Составить процедуру, подсчитывающую сумму положительных и сумму отрицательных элементов одномерного массива. Использовать эту процедуру для подсчета общей суммы положительных и общей суммы отрицательных элементов в двух одномерных массивах.

2. Составить процедуру, печатающую заданную строку символов в обратном порядке. Использовать эту процедуру для преобразования двух заданных строк.

*Вариант 2*

1. Составить процедуру, которая удваивает все нечетные элементы двумерного массива. Используя эту процедуру, преобразовать два двумерных массива.

2. Составить процедуру, проверяющую, что в строке символов имеется баланс открывающих и закрывающих скобок. Использовать эту процедуру для проверки правильности написания двух строк.

*Вариант 3*

1. Составить процедуру формирования матрицы размером (3 х 4), каждый элемент которой получается по формуле

$$X_{i,j} = \frac{i + j}{m}.$$

Сформировать два двумерных массива, используя эту процедуру.

2. Составить процедуру, которая преобразует текст, напечатанный строчными буквами, в текст, напечатанный заглавными буквами. С помощью этой процедуры преобразовать две строки символов.

*Вариант 4*

1. Составить процедуру, которая подсчитывает количество четных и нечетных элементов в одномерном массиве. Использовать эту процедуру для подсчета общего количества четных и общего количества нечетных элементов в двух одномерных массивах.

2. Составить процедуру, проверяющую наличие цифр в строке символов и выдающую сообщение об этом. С помощью этой процедуры проверить две заданные строки.

*Вариант 5*

1. Составить процедуру, которая заменяет все отрицательные элементы двумерного массива их модулями. Используя эту процедуру, преобразовать два двумерных массива.

2. Составить процедуру, которая в строке символов заменяет точку на восклицательный знак. С помощью этой процедуры преобразовать две строки символов.

## Лабораторный практикум № 9. Типизированные файлы. Программирование баз данных на Паскале

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 4.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение основных приемов программирования задач обработки типизированных файлов.

***Вводная часть***

*Определение типизированных файлов баз данных*

К типизированным файлам относятся файлы строго определенного типа. Чаще всего это файлы, состоящие из записей. Они применяются для создания различных баз данных. Стандартное задание такой файловой переменной осуществляется следующим образом:

```
Type
   Filerec = record
                …
              End;
Var
f : File Of Filerec;
```

Примеры описаний типизированных файлов:

```
Type zap=record
Fam: string[30];
Res: integer;
  End;
Var F: file of zap;       //компонентом является запись
```

*Процедуры и функции по работе с типизированными файлами*

Длина любого компонента типизированного файла постоянна, что дает возможность организовать прямой доступ к любому компоненту файла по его порядковому номеру, поэтому типизированные файлы часто называют файлами прямого доступа.

Seek(f, n) – устанавливает файловую переменную f на запись с номером n. Другими словами, процедура смещает указатель файла f на *n*-ю позицию. Нумерация в файле начинается с 0, т.е. номер физической записи на единицу меньше номера логической записи. Это несоответствие в номерах может служить причиной возникновения ошибок чтения-записи.

Для работы с файлами прямого доступа дополнительно можно использовать следующие процедуры и функции.

Определение количества компонентов:

FileSize(f): longint;

Функция возвращает количество компонентов в файле f.

Определение позиции указателя:

FilePos(f): longint;

Функция возвращает порядковый номер текущего компонента файла f.

Отсечение последних компонентов файла:

Truncate(f);

Процедура отсекает конец файла, начиная с текущей позиции включительно.

*Обработка ошибок ввода-вывода*

Компилятор Турбо Паскаля позволяет генерировать выполняемый код в двух режимах: с проверкой корректности ввода-вывода и без нее. При включении проверки в программу ключ режима компиляции обозначается:

{$I+} – режим проверки включен;

{$I-} – режим отключен.

По умолчанию действует режим $I+. Этот ключ компиляции имеет локальную сферу влияния. Можно многократно включать и выключать режим, вставляя в текст программы конструкции {$I+} и {$I-}, тем самым создавая области с контролем ввода–вывода и без него. При включенном режиме проверки любая ошибка ввода–вывода будет фатальной: программа прервется, выдав номер ошибки. Если отключить режим проверки, то при возникновении ошибки ввода-вывода программа не прервется, а продолжит работу со следующего оператора. Функция IOResult : Integer возвращает целое число, соответствующее коду последней ошибки ввода-вывода. Если же операция ввода-вывода прошла без сбоев, то функция вернет значение 0.

Если файл на диске отсутствует, то действие процедуры открытия Reset вызовет ошибку:

```
Assign (f, ‘file1’);
{$I-} Reset (f) {$I+};
If IOResult <> 0      Then Writeln(‘Файл не найден’)
                      Else . . .
```

В этом фрагменте приведен пример того, как можно исключить сбой программы в случае, если файла на диске нет или файл не читается.

***Практическая часть***

***Пример 1***

*Условие задачи*

Создать базу данных, содержащую сведения о служащих: фамилия, должность, оклад. Определить служащего с минимальным окладом и его порядковый номер в файле. Увеличить его оклад на 2000 руб. Организовать вывод содержимого базы данных.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Task;
Type Klerk= Record fam  : String[20];
                   dolgn : String[10];
                   oklad : Real;
            End;
```

```
Var   G: Klerk;              {Переменная типа запись для работы с компонентным файлом}
      f: File Of Klerk;      {Файловая переменная типизированного файла}
      k,i:      Integer;           {Вспомогательные переменные для определения номера
                                   компонента файла}
      FileName: String[20];        {Переменная для имени файла}
      Min: Real;             {Переменная для определения минимального оклада}
Begin
   Write('Введите имя файла');   Readln(FileName);
   Assign(f,FileName);        {Связь файловой переменной с внешним файлом}
                        {***Ввод содержимого базы данных***}
   Rewrite(f);              {Открытие файла на запись}
   With G Do
      Begin
         Write('Фамилия'); Readln(fam);      {Ввод первой фамилии файла сотрудников}
         While Fam<> ' ' Do                   {Пока вместо фамилии не ввели пробел,}
            Begin                           {вводим в цикле очередную запись в файл}
               Write('Должность?'); Readln(dolgn);     {Ввод информации с клавиатуры}
               Write('Оклад?'); Readln(oklad);
               Write(f, G);                          {Запись информации в файл}
               Write('Фамилия'); Readln(fam);
            End;
      End;
   Close(f);
{***Считывание информации из файла и определение номера записи,
          соответствующей минимальному окладу***}
   Reset(f); i:=0;      {Открываем файл для чтения. Начинаем подсчет номера компонента}
   Min := 100000;
   While Not Eof(f) Do        { Организуем цикл для поиска  минимального оклада }
      Begin                    {и порядкового номера компонента с минимальным окладом}
         Read(f,G);              { Считываем очередной компонент файла с диска}
         i:=i+1;
          If G.oklad < Min Then
            Begin Min := G.oklad;
                  k := i;      {запоминаем номер компонента с минимальным окладом}
            End;
       End;
   Seek(f,k-1);                          {Устанавливаем указатель записи перед записью
                                  с минимальным окладом}
   Read(f,G);                    {Считали запись с минимальным окладом с диска}
   Writeln(G.oklad);
   G.Oklad:= G.Oklad + 20;
   Seek(f,k-1);
   Write(f,G);                   {Записали вместо старой записи – новую запись,
                                 с увеличенным окладом}
   Close(f);
                        {***Просмотр содержимого базы данных***}
   Reset(f);   i:=0;
   While Not Eof(f) Do
```

```
        Begin
            Read(f,G);                {Считали компоненту файла с диска}
            i:=i+1;
            Write(i,'  ',G.fam,'  ',G.dolgn,'  ',G.oklad);  {Распечатали содержимое
          Writeln;                                          компонента с номером i}
        End;
    Close(f);
End.
```

***Самостоятельная работа***

*Вариант 1*

Написать программу формирования базы данных, которая содержит информацию об обучающихся: ФИО, номер курса, номер группы, сведения о проживании (дом, общежитие или съемная квартира).

Вывести список обучающихся по определенному месту проживания. Место проживания вводится с клавиатуры.

Изменить в базе данных место проживания обучающейся Иванова А.П. с квартиры на общежитие.

Вывести на экран содержимое измененной базы данных.

*Вариант 2*

Написать программу формирования базы данных, которая содержит сведения за месяц о пропусках занятий обучающихся: ФИО, номер группы, количество часов, пропущенных по уважительной, и количество часов, пропущенных по неуважительной причине.

Вывести на экран содержимое базы данных.

Получить список обучающихся, пропустивших более XXX часов по неуважительной причине. XXX – вводимая величина.

Подсчитать суммарное количество пропусков по неуважительной причине в группе УУУ. УУУ – вводимая величина.

*Вариант 3*

Написать программу формирования базы данных. База данных содержит следующую информацию: фамилия автовладельца, номер автомобиля, марка автомобиля, цвет.

Вывести на экран сведения об автовладельцах (ФИО, номер, цвет) по заданной марке автомобиля.

Изменить в базе данных реквизиты машины Петрова П.П.

Вывести на экран содержимое измененной базы данных.

*Вариант 4*

Написать программу формирования базы данных, которая содержит сведения о сотрудниках: ФИО, величина зарплаты, стаж работы, количество детей.

Получить список сотрудников, стаж которых превышает XXX лет. XXX – вводимая величина

Увеличить выплату сотрудникам, имеющим более четырех детей, на 3000 руб.

Вывести на экран содержимое измененной базы данных.

*Вариант 5*

Создать базу данных, содержащую сведения о товаре: индекс, цена за единицу товара, количество.

Определить индекс товара, цена которого максимальна.

Уменьшить на 50 количество товара с индексом УУУ. УУУ – вводимая величина

Вывести на экран содержимое измененной базы данных.

## Лабораторный практикум № 10. Тестовые файлы. Процедуры работы с тестовыми файлами

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 4.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство со средой программирования PascalABC, изучение основных приемов программирования задач обработки текстовых файлов.

***Вводная часть***

Введение файлового типа в язык Паскаль вызвано необходимостью обеспечить возможность работы с периферийными (внешними) устройствами ЭВМ, предназначенными для ввода, вывода и хранения данных.

Файловый тип данных, или файл, определяет упорядоченную совокупность произвольного числа однотипных компонент.

Понятие файла достаточно широко. Это может быть обычный файл на диске /коммуникационный порт ЭВМ, устройство печати, клавиатура или другие устройства.

При работе с файлами выполняются операции ввода-вывода. Операция ввода означает перепись данных с внешнего устройства (из входного файла) в основную память ЭВМ, операция вывода – это пересылка данных из основной памяти на внешнее устройство (в выходной файл).

Файлы на внешних устройствах часто называют физическими файлами. Их имена определяются операционной системой. В программах на языке Паскаль имена файлов задаются с помощью строк. Например, имя файла на диске может иметь вид

'lab1.dat'

'c:\abc150\pr.txt'

'my_files'

В Паскале существуют три типа файлов:

a) текстовые файлы;

b) типизированные файлы;

c) нетипизированные файлы.

В работе с файлами разных типов имеются общие правила, но существуют и свои особенности. Данный лабораторный практикум посвящен изучению текстовых файлов.

Текстовые файлы – это файлы, компонентами которых являются строки неопределенной длины. Текстовые файлы содержат информацию в виде символов, ее изображающих. Для работы с текстовым файлом необходимо:

а) определить файловую переменную с помощью служебного слова Text;

б) связать файловую переменную с физическим файлом – процедура Assign(<имя файловой переменной>,<имя файла>);

в) инициировать файл, для этого существуют стандартные процедуры;

г) Reset(f) – открывает текстовый файл для чтения;

д) Rewrite(f) – открывает текстовый файл для записи;

е) Append(f) – открывает текстовый файл для добавления данных в конец файла;

ж) информацию в текстовый файл можно записывать операторами Write и Writeln, а считывать – с помощью операторов Read и Readln.

При вводе из текстового файла следует помнить правила чтения значений переменных. Когда вводятся числовые значения, два числа считаются разделенными, если между ними есть хотя бы один пробел, или символ табуляции (#9), или символы конца строки (#13). При чтении информации из текстовых файлов данные преобразуются автоматически. Если при чтении из текстового файла числовой информации символы не могут быть автоматически преобразованы, то возникает ошибка. Пользователь сам должен следить за порядком записи и чтения информации.

Список вывода в операторах Write и Writeln могут составлять константы, переменные целого, вещественного, символьного и строкового типов, а также производные от них. Процедура Write выводит данные в текущую строку и не закрывает ее, т.е. следующие данные запишутся в ту же строку. Процедура

Writeln приписывает символы #13 и #10 в конец строки. Так же, как и при выводе на экран, используется форматный вывод.

Функция Eof(var f) возвращает результат типа boolean: True, если достигнут конец файла, и False – в противном случае. Функция применима также к файлам других типов.

***Практическая часть***

*Условие задачи*

Создать текстовый файл, содержащий фамилии обучающихся. Просмотреть созданный файл. Дополнить файл десятью целыми числами и вновь просмотреть.

Программа на Паскале в этом случае может иметь следующий вид:

```
Program Task;
Var          f : Text;
FileName, Fam : String[20];
I, z : Integer;
{f – файловая переменная; FileName – имя файла; Fam – переменная для ввода/вывода фамилии; i – для параметра цикла; z – переменная для ввода/вывода целых чисел}
Begin
Write('Введите имя файла'); Readln(FileName);
Assign(f,FileName);  {Установка связи между файловой переменной и файлом}
{***Запись фамилий***}
Rewrite(f);   {Открытие файла для записи}
Write('Фамилия?'); Readln(Fam);
Write Fam<>' ' Do {Цикл работает до тех пор, пока вместо ввода фамилии будет нажата
клавиша <Enter>}
Begin
  Writeln(f,Fam);      {Запись информации на диск. После каждой фамилии – символ конца
 строки}
  Writeln('Фамилия ?'); Readln(Fam);   {Очередной ввод с клавиатуры}
End;
Close(f);                                    {Закрытие файла}
{***Просмотр файла***}
Reset(f);                                    {Открытие файла для чтения}
While Not Eof(f) Do                  {Цикл работает до тех пор, пока не достигнут конец файла}
Begin
  Readln(f,Fam);                     {Считываем строку из файла}
  Writeln(Fam);                      {Отражаем строку на экране}
End;
Close(f);
{***Запись 10 целых чисел***}
Append(f);                   {Открытие файла для дополнения}
Writeln(f, '10 целых чисел');          {Запись в файл символьной константы}
For i:=1 To 10 Do
Begin
  Write('Введите целое число'); Read(z);       {Ввод целого числа с клавиатуры}
  Write(f,z,' ');                    {Запись числа на диск. Числа отделены друг от друга пробелом}
End;
Close(f);
{***Просмотр файла***}
{Читаем фамилии из файла и выводим на экран}
```

```
While Fam<>'10 целых чисел' Do
Begin
  Readln(f,Fam);
  Writeln(Fam);
End;
{Читаем числа из файла и выводим на экран}
For i:=1 To 10 Do
Begin
  Read(f,z); Write(z,' ');
End;
Close(f);
End.
```

***Самостоятельная работа***

*Вариант 1*

Создать текстовый файл. Вывести на экран самую длинную строку файла. Если таких строк несколько, вывести на экран все.

*Вариант 2*

Создать текстовый файл, содержащий информацию об обучающихся: фамилия, средний балл (информация разделена признаком конца строки). Создать и просмотреть новый файл, содержащий только фамилии обучающихся из первого файла.

*Вариант 3*

Создать текстовый файл, содержащий фамилии обучающихся. Проверить, что все фамилии в файле начинаются с заглавной буквы. Вывести результаты проверки на экран. Переписать все фамилии из первого файла во второй с исправлением ошибок, если они были допущены. Содержимое второго файла вывести на экран. Исходный файл уничтожить.

*Вариант 4*

Создать текстовый файл. Подсчитать среднюю длину строк этого файла. Переписать все строки, длина которых превосходит среднее значение, в другой файл. Вывести на экран второй файл.

*Вариант 5*

Создать текстовый файл, содержащий сведения о людях: фамилия, возраст, пол (информация разделена признаком конца строки). Создать два других текстовых файла, в один из которых переписать фамилии всех женщин из исходного файла, во второй – всех мужчин. Просмотреть созданные текстовые файлы.

## Лабораторный практикум № 11. ССЫЛОЧНЫЕ ПЕРЕМЕННЫЕ

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 4.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** научить пользователя применять ссылочные переменные для создания списков, стеков, деревьев.

***Вводная часть***

Описание ссылочного типа выглядит следующим образом:

Type Ptr = ^t;

где t – стандартный или заранее описанный тип данных, называемый базовым типом. Сами адреса будут храниться в ссылочных переменных, которые описываются обычным образом, например, Var P : Ptr. Такие переменные называются ссылками или указателями.

Список – это набор записей, каждая из которых имеет поле данных и указатель (ссылку) на следующую запись в списке. Та, в свою очередь, тоже содержит поле данных и ссылку на продолжение списка. Последний элемент списка содержит значение Nil, т.е. уже ни на что не ссылается. Начало списка формирует переменная типа "указатель", содержащая адрес первого элемента списка. Поле данных еще называют информационной частью списка.

Пример описания элемента списка, информационная часть которого – переменная типа Integer:

```
Type lst_ptr = ^lst_type;
     lst_type = record
        data : integer;
        next : lst_ptr;
     End;
```

Список обычно задается указателем на свой первый элемент. Пусть это указатель p.

Добавление элемента в начало списка можно осуществить при помощи следующей последовательности операторов:

```
New(p);                 {Выделение памяти для элемента списка}
Readln(q^.data);        {Задаем информационную часть списка}
q^.next:=p;             {Связали элемент с остальными элементами списка}
p:=q;                   {Перенесли указатель на начало, т.е. на добавленный элемент}
```

Основными операциями над списками являются:

- переход от элемента к следующему элементу;
- включение нового элемента в список;
- исключение элемента из списка.

Стек – линейный список, в котором добавления и исключения элемента производятся с одного конца, называемого вершиной стека.

Работа со стеком осуществляется через указатель стека. При выполнении загрузки элемента в стек данные записываются на место, определяемое указателем стека, а указатель стека изменяет свое состояние и задает следующую свободную ячейку блока памяти. При извлечении элемента из стека указатель стека возвращается назад на один шаг.

***Практическая часть***

Фрагмент программы, в котором формируется список целых чисел, полученных с помощью датчика случайных чисел:

```
p:=nil;
For i:=1 To 10 Do
Begin
  New(q);
  q^.data:=random(100);
  q^.next:=p;
  p:=q;
End;
```

*Процедура обхода элементов списка и вывода содержимого каждого элемента на экран:*

```
Procedure pass_l(L : lst_ptr);
{Входной параметр: L типа lst_pte – указатель на первый элемент списка}
Var p : lst_ptr;
Begin
  p:=L;                     {Встать на начало списка}
  While p<>nil Do    {Пока не конец списка}
  Begin
    Writeln(p^.data);  {Вывести содержимое списка}
    p:=p^.next;             {Перейти к следующему элементу списка}
```

```
  End;
End;
```
*Процедура поиска в списке элемента, содержащего искомые данные:*
```
Procedure seach(L : lst_ptr; key : integer; Var p ; lst_ptr);
{Входные параметры:
L - типа lst_ptr - указатель на начало списка, в котором производится поиск;
key - значение данных, которое должен содержать искомый элемент (ключ поиска).
Выходной параметр:
p - типа lst_ptr - указатель на элемент списка, содержащего искомые данные,
в случае отсутствия такого элемента равен nil}
Begin
  p:=L;
  While (p<>nil) and (p^.data<>key) Do p:=p^.next
End;
```
*Процедура, добавляющая в список элемент p после элемента q:*
```
Procedure add_after(p,q : lst_ptr);
{Входные параметры:
p типа lst_ptr - указатель на вставляемый элемент,
q типа lst_ptr - указатель на элемент списка, после которого вставляется элемент p}
Begin
  p^.next:=q^.next;
  q^.next:=p;
End;
```
*Процедура удаления из списка элемента p:*
```
Procedure del(p : lst_ptr; Var L : lst_ptr);
{Параметры: p - ссылка на удаляемый элемент; L – ссылка на первый элемент списка}
Var q : lst_ptr;
{q – рабочий указатель, который будет ссылаться на элемент, предшествующий p}
Begin
  If p<>L Then         {если p – не первый элемент списка}
  Begin
     q:=L;     {поиск элемента, предшествующего p}
     While q^.next <> p Do q:=q^.next;
     q^.next:=p^.next;
  End
  Else          {Обработка случая, когда удаляется первый элемент списка}
     L:=p^.next;
End;
```
*Добавить элемент в стек:*
```
New(q); Readln(q^.x);  q^.next:=p; p:=q;
```
*Считать значение элемента из стека и исключить его из стека:*
```
q:=p;  p:=q^.next;  y:=q^.x;  dispose(q);
```
*Считать элемент, не удаляя его из стека:*
```
y:=p^.x;
```
***Самостоятельная работа***

*Вариант 1*

Напишите программу, которая:

1) создает список из 20 символьных элементов;

2) создает второй список, содержащий копию первого списка, но в обратном порядке.

*Вариант 2:*

1) напишите программу, которая создает список женских имен;

2) исключите из списка элемент, содержащий имя XXX. XXX – вводимая величина.

*Вариант 3:*

1) напишите программу формирования списка случайных целых;

2) удалите из списка все элементы, содержащие в качестве части числа, большие УУУ;

3) подсчитайте среднее арифметическое оставшихся элементов УУУ – вводимая величина.

*Вариант 4:*

1) напишите программу формирования списка целых величин;

2) из исходного списка необходимо сформировать два новых списка: один из положительных элементов, а другой – из остальных элементов списка.

*Вариант 5:*

1) напишите программу формирования списка целых величин;

2) подсчитайте количество вхождений заданного элемента E в сформированный список;

3) замените в списке все вхождения элемента E1 на E2 (E, E1 и E2 вводятся с клавиатуры).

**Лабораторный практикум № 12. Программирование алгоритмов линейной, разветвляющейся и циклической структуры на языке C++**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 7.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** знакомство с интегрированной средой разработки Visual Studio, с программированием линейных, разветвляющихся и циклических алгоритмов на языке программирования C++.

***Вводная часть***

*Описание интегрированной среды разработки* Visual Studio

Visual Studio – интегрированная среда разработки – это программный продукт, объединяющий текстовый редактор, компилятор, отладчик и справочную систему.

Любая программа в среде Visual Studio всегда создается в виде отдельного проекта. Проект (project) – это набор взаимосвязанных исходных файлов и, возможно, включаемых заголовочных файлов, компиляция и компоновка которых позволяет создать исполняемую программу. Основу Visual Studio составляет рабочая область (project workspace). Она может содержать любое количество различных проектов, сгруппированных вместе для согласованной разработки: от отдельного приложения до библиотеки функций или целого программного пакета. Решение же простых задач сводится к оформлению каждой программы в виде одного проекта, т.е. рабочая область проекта будет содержать ровно один проект.

При запуске главное окно интегрированной среды Visual Studio имеет следующий вид (рисунок 98).

Экран Visual Studio разделен на четыре основные зоны (рисунок 99). Сверху расположены меню и панели инструментов. Кроме них рабочий стол Visual Studio включает в себя три окна:

Окно Project Workspace (окно рабочей области) – расположено в левой части. Первоначально окно закрыто, но после создания нового проекта или загрузки существующего проекта это окно будет содержать несколько вкладок.

Окно Editor (окно редактирования) расположено справа. Его используют для ввода, проверки и редактирования исходного кода программы.

Окно Output (окно вывода) служит для вывода сообщений о ходе компиляции, сборки и выполнения программы и сообщений о возникающих ошибках.

Вкладки окна Workspace представляют собой инструменты для просмотра (Viewers) файлов проекта:

• ClassView – демонстрирует классы, их данные и методы;

• FileView – показывает все файлы, включенные в проект;

• ResourceView – позволяет быстро попасть в нужный редактор ресурсов.

При создании нового проекта (по команде меню File – New-Project) по умолчанию создается один проект, но в двух конфигурациях:

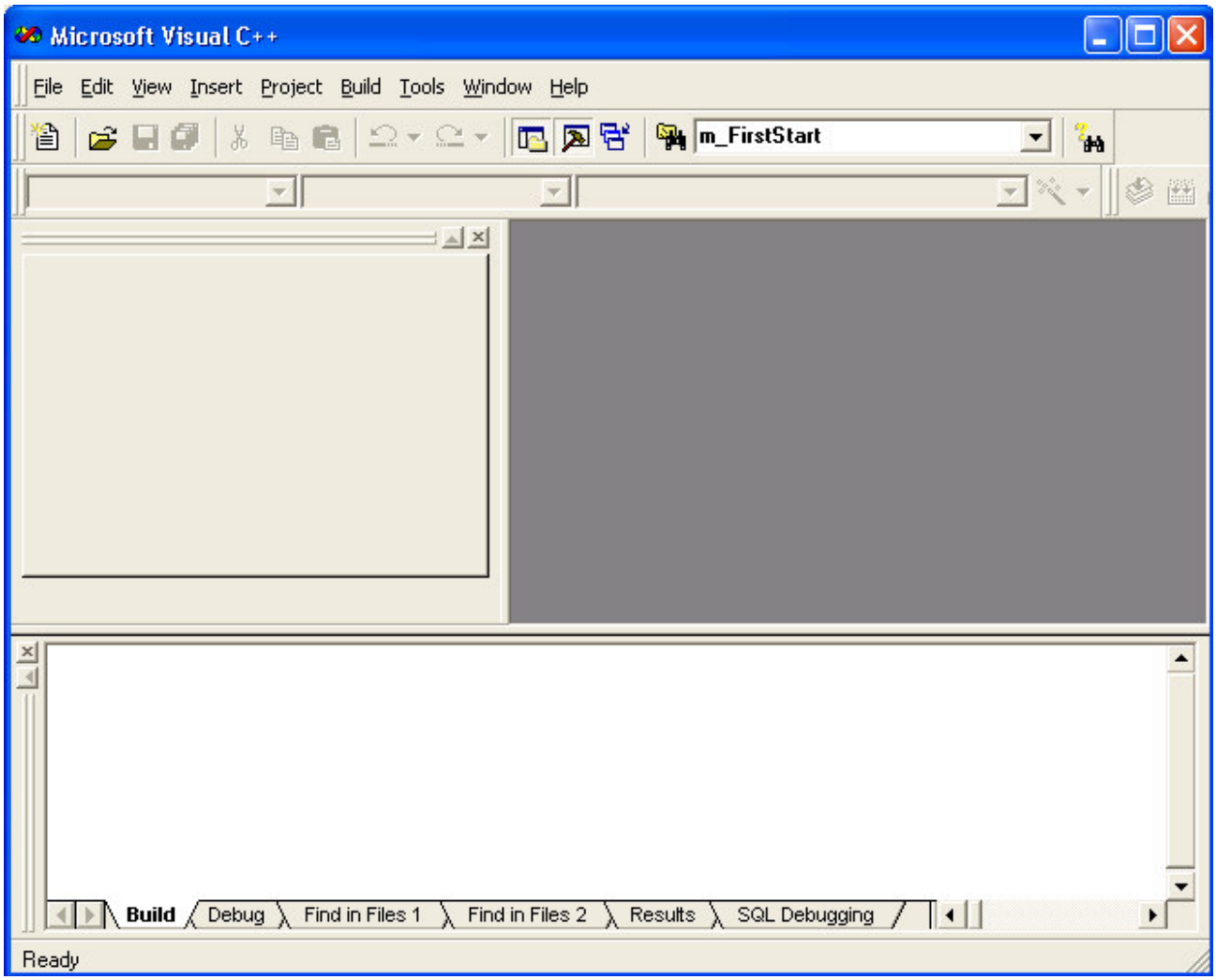



Рисунок 98. Главное окно интегрированной среды Visual C++

• Debug – версия проекта, в которой выключается оптимизация кода и включается в него отладочная информация;

• Release – версия, в которой делается все наоборот (выключается отладочная информация и включается оптимизация кода).

Сначала отображается только одна из папок (Debug или Release) в зависимости от выбора активной конфигурации в меню Build – Set Active Configuration. Обычно при создании нового проекта (Project) он автоматически помещается во вновь созданное рабочее пространство (Workspace) в конфигурации Debug. Когда создано рабочее пространство, в него можно добавлять:

• новые проекты;

• новые конфигурации (configurations);

• взаимозависимости (interdependencies) между проектами;

• подчиненные проекты (subprojects).

Visual Studio предлагает выбор из семнадцати различных типов проектов приложений. Их все можно увидеть, выбрав пункт меню File – New-Project. Например:

• Win32 Application – проект Windows-приложения, ориентированного на использование функций API и использующего GDI – Graphic Device Interface (графический интерфейс Windows);

• Win32 Console Application – проект приложения, не использующего GDI (текстовый режим окна);

• MFC AppWizard (exe) – проект приложения для Windows с использованием библиотеки MFC (Microsoft Foundation Classes), основанный на начальном остове или стартовой заготовке приложения.

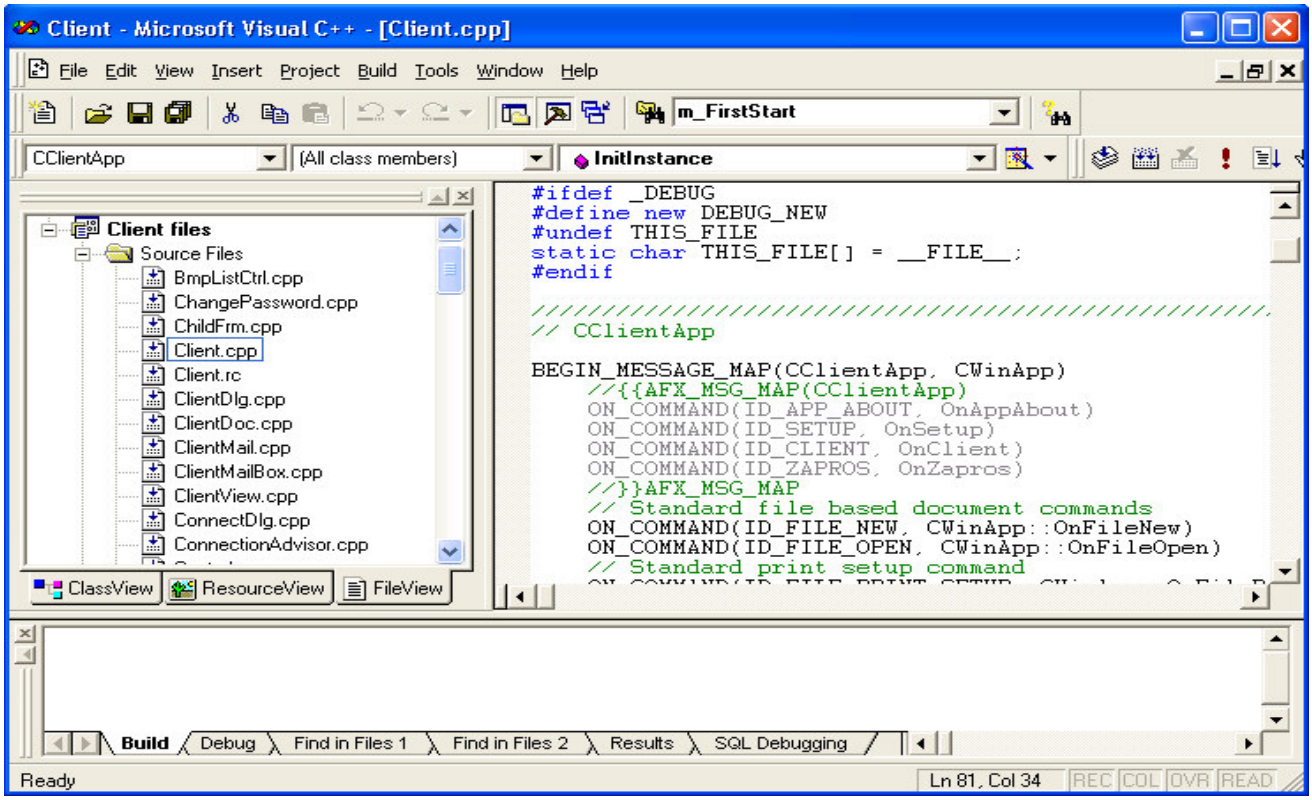



Рисунок 99. Главное окно интегрированной среды Visual C++

Назначение основных команд меню среды Visual Studio:

• в меню File собран стандартный для многих приложений Windows набор команд, предназначенных для манипулирования файлами;

• команды меню Edit позволяют редактировать текст и проводить поиск по ключевым словам в программном коде, отображаемом в активном окне; работа этих команд основана на тех же принципах, что и работа аналогичных команд в большинстве текстовых редакторов;

• меню View содержит команды, позволяющие настроить внешний вид рабочего пространства;

• меню Insert содержит команды, позволяющие вставлять в проект новые файлы, ресурсы, объекты и т.д.;

• команды меню Project позволяют управлять открытыми проектами;

• в меню Build содержатся всевозможные команды, предназначенные для генерации кода приложения, отладки и запуска программы;

• меню Tools содержит команды вызова вспомогательных утилит, программирования макросов и настройки среды Visual Studio;

• команды меню Window, за исключением команды DockingView (с помощью этой команды можно закрепить панель инструментов у любого края родительского окна либо сделать ее свободно перемещаемой), соответствуют стандартному набору команд данного меню во всех приложениях Windows;

• меню Help содержит стандартные для приложений Windows команды Contents, Search и Index, а также некоторые дополнительные команды.

*Структура программы в C++*

В общем случае программа C++ состоит из нескольких блоков или частей:

1. Блок заголовков программы. Обычно в этом блоке с помощью инструкции #include подключаются внешние файлы.

2. Блок с объявлением классов (базовых и производных), прототипами и объявлениями функций.

3. Главный метод программы: каждая программа имеет такой метод. У метода стандартное название main().

4. Блок с описанием функций (прототип которых указан во втором блоке).

При этом обязательными являются только первый и третий блоки: программа содержит блок подключения файлов и главный метод main().

Метод – это фактически синоним слова функция (или процедура). Запуск программы означает выполнение ее главного метода main (). У программы может быть один и только один метод main().

Особенность языка С++ состоит еще и в том, что этот язык в определенном смысле переходной – он позволяет создавать как объектно-ориентированные программы, так и программы, подобные тем, что характерны для процедурных языков программирования.

*Объявление и инициализация переменных в С++*

В общем случае объявление переменной в С++ осуществляется путем указания типа переменной и ее имени. Синтаксис объявления переменной имеет вид:

тип имя переменной

Например:

int age;

Идентификатором типа является инструкция int, а именем переменной age. Всего в С++ есть семь базовых типов (таблица 9).

Таблица 9. Базовые типы С++

| Идентификатор типа | Тип данных |
|---|---|
| bool | Логический тип |
| char | Символьный тип |
| wchar_t | Символьный двухбайтовый тип |
| double | Действительные числа двойной точности |
| float | Действительные числа |
| int | Целые числа |
| void | Значение не возвращается |

Вместе с идентификаторами типов могут использоваться так называемые модификаторы типа. Модификаторы типа – это специальные ключевые слова, которые указываются перед идентификатором типа и позволяют изменять базовый тип. В С++ используются модификаторы signed (значения со знаком), unsigned (значения без знака), short (укороченный тип) и long (расширенный тип). Все четыре модификатора могут использоваться для типа int. Модификаторы signed и unsigned, кроме этого, используются с типом char, а с типом double используют модификатор long.

Что касается диапазона значений для данных разных типов, то в разных компиляторах диапазоны значений данных различны. В языке С++ вводятся стандарты только для минимально необходимого диапазона, который должен поддерживать компилятор.

Ключевое слово void используется при определении функций, которые не возвращают результата. Это функции – аналог процедур в таких языках программирования, как, например, Pascal. Ключевое слово void также используется при определении обобщенных указателей.

Если переменная объявлена как такая, что принадлежит к определенному типу, в дальнейшем изменить ее тип невозможно. Объявляются переменные фактически в любом месте программного кода, однако использовать в программе переменную можно только после того, как ей присвоено значение. В этом случае говорят об инициализации переменной. Инициализация переменной также может выполняться в любом месте программы, но не ранее объявления этой переменной и до места первого ее использования. Объявление и инициализацию переменной часто совмещают. Например, инструкцией int: n=10 объявлена целочисленная переменная n со значением 10. Если объявляется несколько переменных одного типа, достаточно перечислить эти переменные, разделенные запятой, после идентификатора типа. То же самое касается инициализации переменных при объявлении. Например:

```
// Объявление трех целочисленных переменных n, m и k:
int n, m, k;
// Объявление с одновременной инициализацией нескольких переменных:
int one=1, two=2, three, four=4, five;
```

Однако объявлять переменные лучше в начале соответствующего программного блока, причем желательно указывать персонально для каждой переменной идентификатор типа. Программный код в таком случае пользователем воспринимается намного лучше.

В C++ существует возможность при инициализации переменных использовать не только литералы (явно указанные значения, в соответствии с типом переменной), но и выражения, в которые входят другие переменные. Единственное условие состоит в том, чтобы выражение, на основании которого инициализируется такая переменная, было легитимным на момент инициализации.

*Арифметические, логические, поразрядные и операторы отношения C++*

Выражения в C++ содержат, помимо переменных, еще и операторы. Операторы условно можно поделить на арифметические, логические, поразрядные и операторы отношения.

*Арифметические операторы*

Арифметические операторы используются для сложения, вычитания, умножения и деления чисел. Список основных арифметических операторов, которые используются в C++, представлены в таблице 10.

Таблица 10. Арифметические операторы, используемые в C++

| Оператор | Назначение |
|---|---|
| + | Сложение |
| - | Вычитание |
| * | Умножение |
| / | Деление. Если операндами являются целые числа, выполняется целочисленное деление |
| % | Остаток от деления (деление по модулю) |
| ++ | Инкремент |
| -- | Декремент |

Первые пять операторов являются бинарными, то есть используются с двумя операндами.

Операторы инкремента и декремента являются унарными (используются с одним операндом). Действие операторов состоит в увеличении или уменьшении значения операнда на единицу. Например, результат выполнения команды i++ есть увеличение значения переменной i на единицу. Другими словами, команды i++ и i=i+1 с точки зрения влияния на значение переменной i являются эквивалентными. Аналогично в результате команды

i-- значение переменной i уменьшается на единицу, как и в результате выполнения команды i=i-1.

Операторы инкремента и декремента могут использоваться в префиксной и постфиксной формах. В префиксной форме оператор указывается перед операндом, а в постфиксной форме – после него. Выше операторы инкремента и декремента использовались в постфиксной форме. В префиксной форме соответствующие операции выглядели бы как ++i и --i.

В плане действия на операнд разницы в префиксной и постфиксной формах нет. В результате выполнения команды ++i значение переменной i увеличивается на единицу, как и для команды i++. Уменьшение значения переменной i на единицу осуществляется при использовании команды --i. В этом отношении она не отличается от команды i--. Разница между постфиксной и префиксной формами операторов инкремента и декремента проявляется в ситуации, когда эти операторы использованы в выражениях. Естественным образом возникает вопрос относительно переменной-операнда, по отношению к которой применяется операция инкремента или декремента и которая является составной частью более сложного выражения: следует ли сначала вычислить выражение и затем изменить значение переменной или сначала следует изменить значение переменной, а уже потом вычислять выражение? Ответ следующий: для префиксной формы сначала изменяется значение

переменной, а затем рассчитывается выражение, а для постфиксной формы выражение вычисляется со старым значением переменной, а только после этого изменяется значение этой переменной.

*Логические операторы*

Логические операторы предназначены для работы с операндами логического типа, и результатом соответствующих операций являются значения логического типа. В C++ всего три логических оператора (таблица 11).

Таблица 11. Логические операторы, используемые в C++

| Оператор | Назначение |
|---|---|
| && | Логическое И. Бинарный оператор. Результатом выражения A && B является true, если оба операнда A и B равны true. Результатом выражения A && B является false, если хотя бы один из операндов A или B равен false |
| \|\| | Логическое ИЛИ. Бинарный оператор. Результатом выражения A \|\| B является true, если хотя бы один из операндов A или B равен true. Результатом выражения A \|\| B является false, если оба операнда A и B равны false |
| ! | Логическое отрицание. Унарный оператор. Результатом выражения !A является значение true, если операнд A равен false. Если операнд A равен true, значение выражения ! A равно false |

*Операторы сравнения*

Операторы сравнения используются для сравнения значений операндов. Результатом выражения на основе оператора сравнения является логическое значение true, если соответствующее условие выполнено, и false – в противоположном случае. Операторы сравнения перечислены в таблице 12.

Таблица 12. Операторы сравнения, используемые в C++

| Оператор | Назначение |
|---|---|
| > | Больше |
| < | Меньше |
| >= | Больше или равно |
| <= | Меньше или равно |
| == | Равно |
| != | Не равно |

Все перечисленные операторы являются бинарными.

*Побитовые операторы*

Язык программирования C++ обладает полным набором побитовых операторов. Побитовые операторы применяются при выполнении операций с битами в двоичном представлении числовых значений (таблица 13).

Таблица 13. Побитовые операторы, используемые в C++

| Оператор | Назначение |
|---|---|
| & | Побитовое И. Бинарный оператор. Результатом выражения *a* & *b* является число, каждый бит которого в двоичном представлении равен результату сравнения соответствующих битов чисел a и b: значение бита равно 1, если оба сравниваемых бита равны 1. В противном случае значение бита равно 0 |
| \| | Побитовое ИЛИ. Бинарный оператор. Результатом выражения *a* \| *b* является число, каждый бит которого в двоичном представлении равен результату сравнения соответствующих битов чисел *a* и *b*: значение бита равно 1, если хотя бы один из сравниваемых битов равен 1. В противном случае значение бита равно 0 |
| ^ | Побитовое исключающее ИЛИ. Бинарный оператор. Результатом выражения *a* ^ *b* является число, каждый бит которого в двоичном представлении равен результату сравнения соответствующих битов чисел *a* и *b*: значение бита равно 1, если один и только один из сравниваемых битов равен 1. В противном случае значение бита равно 0 |
| ~ | Побитовое отрицание (дополнение до единицы). Унарный оператор. Результатом выражения ~*a* является число, которое получается побитовым инвертированием числа *a* |
| >> | Сдвиг вправо. Бинарный оператор. В двоичном представлении числа, указанном |

| | |
|---|---|
| | слева от оператора, выполняется сдвиг всех битов вправо на число позиций, указанных справа от оператора. При этом старший бит знака остается неизменным, а выходящие за диапазон младшие биты теряются |
| << | Сдвиг влево. Бинарный оператор. В двоичном представлении числа, указанном слева от оператора, выполняется сдвиг всех битов влево на число позиций, указанных справа от оператора, с заполнением младших битов нулями и потерей старших битов |

Пример: 5 & 3. В двоичном представлении число 5 имеет вид 101, а число 3 представляется как 011. Побитовое сравнение чисел 101 и 011 с помощью оператора побитового И & дает 001, что в десятичной системе соответствует числу 1.

***Условные операторы***

*Тернарный оператор*

В C++ есть тернарный оператор (у оператора три операнда), который позволяет в зависимости от некоторого условия (первый операнд) выполнять различные действия (второй и третий операнды). Синтаксис вызова оператора:

условие ? выражение1 : выражение2

Фактически тернарный оператор представляет собой сокращенную форму условного оператора.

Тернарный оператор возвращает значение. Сначала проверяется указанное первым операндом условие. Если условие выполнено, вычисляется выражение1 после вопросительного знака. Если условие не выполнено, вычисляется выражение2 после двоеточия. Тернарным оператором в качестве значения возвращается значение вычисленного выражения.

Например, командой n>0?5.4:3.2 проверяется условие n > 0, и если это так, возвращается значение 5.4. В противном случае возвращается значение 3.2.

*Условный оператор if()*

Оператор if() позволяет выполнять разные блоки операторов в зависимости от того, выполняется ли некое условие. Условие указывается в круглых скобках после ключевого слова if. Общий синтаксис вызова оператора следующий:

if (условие) {операторы 1} else {операторы 2}

Если условие, указанное после ключевого слова if, верно, выполняется блок операторов операторы 1. В противном случае выполняется блок операторов операторы 2, указанных после ключевого слова else. После выполнения условного оператора управление передается оператору, следующему после него.

Допускается использование упрощенного варианта условного оператора, в котором отсутствует ветка else для выполнения операторов при невыполнении условия. Синтаксис вызова условного оператора в такой форме имеет вид

if (условие) {операторы 1}

В этом случае при выполнении условия управление передается блоку операторов, указанному после ключевого слова if. Если условие не выполнено, выполняются операторы, размещенные после условного.

Синтаксис вызова вложенных условных операторов:

```
if (условие 1) {операторы 1}
        else if (условие 2) {операторы 2}
                else if (условие 3) {операторы 3}
                                ……
                                        else if (условие N) {операторы N}
                                                else {операторы N+1}
```

*Условный оператор switch()*

В тех случаях, когда проверяется больше одного условия, вместо нескольких вложенных условных операторов if() нередко используют оператор switch(). Синтаксис вызова оператора switch() следующий:

switch (выражение) {

```
case значение1:
операторы
break;
case значение2:
операторы
break;
   …
   default:
операторы
}
```

В круглых скобках после ключевого слова switch указывается выражение, значение которого проверяется. Результатом выражения может быть целое число или символ. Значение, возвращаемое выражением, сравнивается со значениями, указанными после ключевых слов case. Если имеет место совпадение, выполняется соответствующий блок операторов. Операторы выполняются до конца оператора switch() или пока не встретится инструкция break (в общем случае инструкция break используется для выхода из оператора цикла и перехода к следующему оператору). Если совпадения нет, выполняются операторы после инструкции default.

***Операторы цикла***

Операторы цикла позволяют многократно выполнять серии однотипных действий. Действия выполняются до тех пор, пока остается справедливым (или пока не будет выполнено) некоторое условие.

*Оператор цикла for()*

Общий синтаксис вызова оператора for() следующий:

for(инициализация; условие; изменение переменных) {команды}

В круглых скобках после ключевого слова for указывается программный код из трех блоков (при этом каждый из блоков может быть пустым). Блоки разделяются точкой с запятой. Первый блок является блоком инициализации. В нем обычно присваиваются начальные значения для переменной (или переменных) цикла. Второй блок – условие выполнения оператора цикла. Пока справедливо условие, оператор цикла будет выполняться. Третий блок – это блок изменения индексных переменных. Указанное назначение блоков достаточно условное. Общий принцип выполнения оператора цикла: сначала выполняются команды, указанные в первом блоке оператора for(). После этого проверяется условие, указанное во втором блоке оператора. Если условие справедливо, выполняются команды после инструкции for() (если команд несколько, они заключаются в фигурные скобки). После выполнения команд в фигурных скобках выполняются команды третьего блока в круглых скобках после ключевого слова for. Далее снова проверяется условие (второй блок). При справедливости условия снова выполняются команды в фигурных скобках и команды третьего блока и т.д. Схема выполнения оператора цикла представлена на рисунке 100.

Пример расчета суммы натуральных чисел:

```
#include <iostream.h>
void main() {
   int n,i,s=0;
   cout<<"n = "; cin>>n;
   for(i=1; i<=n; i++) {
            s+=i;
   }
   cout<<"s = "<<s<<"\n";
}
```

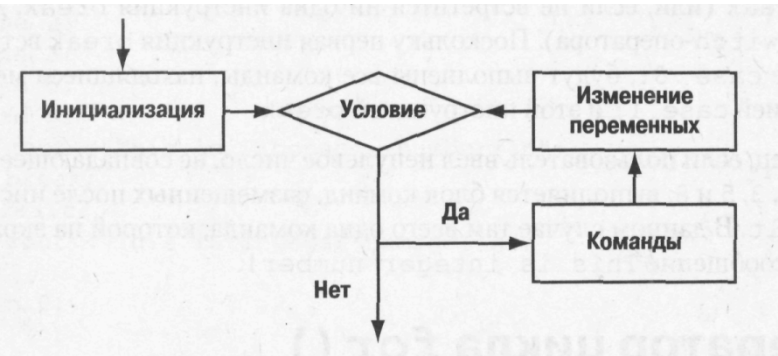



Рисунок 100. Схема выполнения оператора цикла

Основу программы составляет оператор цикла for(i=1; i<=n; i++) { s+=i;}, который содержит в первом блоке команду инициализации индексной переменной i=1 с начальным единичным значением. Второй блок – проверяемое условие i<=n. Это означает, что оператор цикла выполняется до тех пор, пока индексная переменная i не превышает значения переменной n (значение переменной предварительно вводится с клавиатуры). В третьем блоке указана инструкция i++, в силу чего значение индексной переменной увеличивается на единицу. Наконец, в основном блоке оператора цикла (в фигурных скобках) использована команда s+=i, которой на каждом шаге целочисленная переменная s (начальное нулевое значение этой переменной установлено при ее объявлении) увеличивается на значение индексной переменной i. Сразу отметим, что поскольку в основном блоке оператора цикла команда всего одна, фигурные скобки можно было не использовать.

Процедура выполнения оператора цикла следующая. Сначала индексной переменной присваивается единичное значение. Затем проверяется условие, и если индексная переменная меньше значения n, выполняется команда s+=i (с текущими значениями переменных s и i). Далее выполняется команда i++, что приводит к увеличению на единицу индексной переменной, после чего снова проверяется условие. Процесс будет продолжаться до тех пор, пока значение индексной переменной не превысит значения переменной n. Таким образом, после выполнения оператора цикла значение переменной s определяется суммой натуральных чисел от 1 до n. Это значение выводится на экран.

Некоторые (а то и все) блоки в круглых скобках после ключевого слова for могут быть пустыми. Например, при составлении программы допускается использовать и несколько иной синтаксис вызова оператора цикла. Например:

```
#include <iostream.h>
void main() {
    int n,i=1,s=0;
    cout<<"n = "; cin>>n;
    for(; i<=n; i++) {
            s+=i;
    }
    cout<<"s = "<<s<<"\n";
}
```

Индексная переменная в операторе цикла - понятие достаточно условное. Обычно под такой переменной подразумевают целочисленную переменную, которая пробегает набор дискретных значений. Однако это не всегда так. В листинге приведен пример программы, в которой роль индексной переменной выполняет переменная типа char:

```
#include <iostream.h>
void main() {
    for(char x='a'; x!='z';) {
```

```
            cout<<"Guess symbol: "; cin>>x;
    }
cout<<"Correct!\n";
}
```

Основу программы составляет оператор цикла с заголовком for(char x='a'; x!='z';), в котором переменная x типа char инициализируется со значением 'a'. Проверяемым условием является x!='z', что означает продолжение оператора цикла до тех пор, пока значение переменной x не станет равным 'z'. Третьего блока нет. Изменение значения переменной x происходит в результате считывания с клавиатуры. Таким образом, на экране будет отображаться фраза "Guess symbol:" (угадайте букву), после чего пользователь должен ввести букву. Процесс продолжается до тех пор, пока не будет введена буква z. После этого отображается фраза "Correct!" (правильно).

*Оператор цикла while()*

Помимо оператора цикла for() , широко используются циклы while() и do-while(). Синтаксис вызова оператора while() следующий:

```
while (условие) {
команды
}
```

Сначала проверяется условие, указанное в круглых скобках после ключевого слова while. Если условие справедливо, поочередно выполняются операторы, указанные в фигурных скобках после инструкции while(). Если инструкция одна, фигурные скобки можно не указывать.

Синтаксис вызова оператора do-while() имеет вид:

```
do {
команды
} while (условие);
```

В операторе цикла do-while() выполняемые команды (заключенные в фигурные скобки) указываются после ключевого слова do. Далее проверяется условие, указанное в круглых скобках после ключевого слова while. Если условие выполнено, снова выполняются команды после ключевого слова do и т.д.

Таким образом, принципиальная разница между операторами while() и do-while() состоит в том, что в первом случае сначала проверяется условие, а затем (если верно условие) выполняются команды. Во втором случае сначала, по крайней мере, один раз выполняются команды, а затем проверяется условие. По сравнению с оператором цикла for() операторы while() и do-while() требуют от программиста большей ответственности в первую очередь в плане детальной проработки механизма изменения значения проверяемого условия в процессе выполнения команд основного блока оператора. Программный код должен быть составлен корректно, чтобы не получить бесконечный цикл.

В листинге представлена реализация программы для вычисления суммы натуральных чисел с помощью оператора цикла while().

```
#include <iostream.h>
void main() {
    int n,i=1,s=0;
    cout<<"n = "; cin>>n;
    while(i<=n) {
            s+=i;
            i++;
    }
    cout<<"s = "<<s<<"\n";
}
```

В принципе, вместо двух команд s+=i и i++ можно было использовать одну команду s+=i++, но в данном случае это не принципиально. Алгоритм выполнения программы следующий: сначала выводится текстовое

сообщение с приглашением ввести число. После ввода пользователем числа проверяется неравенство i<=n (переменная i предварительно инициализирована с единичным значением). Если значение переменной i не превышает введенного пользователем значения n, выполняются команды оператора цикла (s+=i и i++, соответственно). После этого снова проверяется условие и т.д. до тех пор, пока значение переменной i не превысит значения n. Результат вычисления суммы чисел выводится на экран.

Пример использования оператора do-while() приведен в листинге:

```
#include <iostream.h>
void main() {
    int n,i=1,s=0;
    cout<<"n = "; cin>>n;
    do {
            s+=i;
            i++;
    } while(i<=n);
    cout<<"s = "<<s<<"\n";
}
```

***Инструкция безусловного перехода***

В C++ существует инструкция goto, которая позволяет выполнять переход к заранее определенному месту программы. Место, к которому осуществляется переход, помечается с помощью специального идентификатора, который называют меткой. Чтобы вставить в программный код метку, необходимо ввести в соответствующем месте имя метки с двоеточием в конце. Чтобы перейти к помеченному месту кода, необходимо после инструкции goto указать метку, определяющую место перехода.

Использование инструкций безусловного перехода в программах считается дурным тоном. Существует точка зрения, что наличие в программе инструкций goto замедляет процесс выполнения программы и понижает читабельность программного кода.

Работу с инструкциями безусловного перехода проиллюстрируем на примере создания цикла для расчета суммы чисел. В листинге приведен код программы, в которой вместо оператора цикла использована конструкция с меткой, условным оператором и инструкцией безусловного перехода:

```
#include <iostream.h>
void main() {
    int n,i=1,s=0;
    cout<<"n = "; cin>>n;
    mylabel:   // используется метка
    s+=i;
    i++;
    if (i<=n) goto mylabel;
    cout<<"s = "<<s<<"\n";
}
```

Начальная часть программы ничем не отличается от рассмотренных ранее вариантов: объявляются целочисленные переменные i (начальное значение 1), s (начальное значение 0) и n (значение вводится пользователем). После метки mylabel следует три команды: s+=i, i++ и if (i<=n) goto mylabel. Именно с помощью этих команд реализуется цикл. Первые две особых комментариев не требуют. Третьей командой в рамках условного оператора проверяется условие i<=n. Если значением выражения является true, выполняется безусловный переход к тому месту в программе, что выделено меткой mylabel, т.е. к началу рассмотренного блока. Процесс продолжается до тех пор, пока значение переменной i не превысит значение переменной n. В этом случае в условном операторе команда безусловного перехода не выполняется, а управление передается к следующему после условного оператору.

***Практическая часть***

*Задача 1.* Разработать приложение, вычисляющее координаты тела, брошенного под углом к горизонту.

Если телу в начальный момент сообщена скорость v и тело брошено под углом α к горизонту, то зависимость x-координаты от времени t дается соотношением $x(t) = v\cos(\alpha)\,t$. Для y-координаты закон движения имеет вид $y(t) = v\sin(\alpha)\,t - \frac{gt^2}{2}$, где $g \approx 9.8$ м/с2 есть ускорение свободного падения. Время полета тела до падения составляет $T = 2v\sin(\alpha)/g$. В программе вводится значение скорости тела и угол, под которым тело брошено к горизонту. Угол вводится в градусах, поэтому вводимое значение переводится в радианы (умножается на π≈3.1415 и делится на 180). Далее вычисляется время полета тела, и пользователю предлагается ввести момент времени (не превышающий время полета тела), для которого необходимо рассчитать координаты тела.

1. Создайте рабочую папку Work.
2. Запустите Microsoft Visual Studio.
3. Выполните команду File – New… Выберите вкладку Projects.
4. В поле Location установите рабочую директорию. В поле Project name введите имя проекта Task1. Выберите тип создаваемого приложения – Win32 Console Application (рисунок 101).
5. Нажмите на кнопку OK.
6. Выберите тип An empty project (пустой проект) и нажмите на кнопку Finish (рисунок 102).

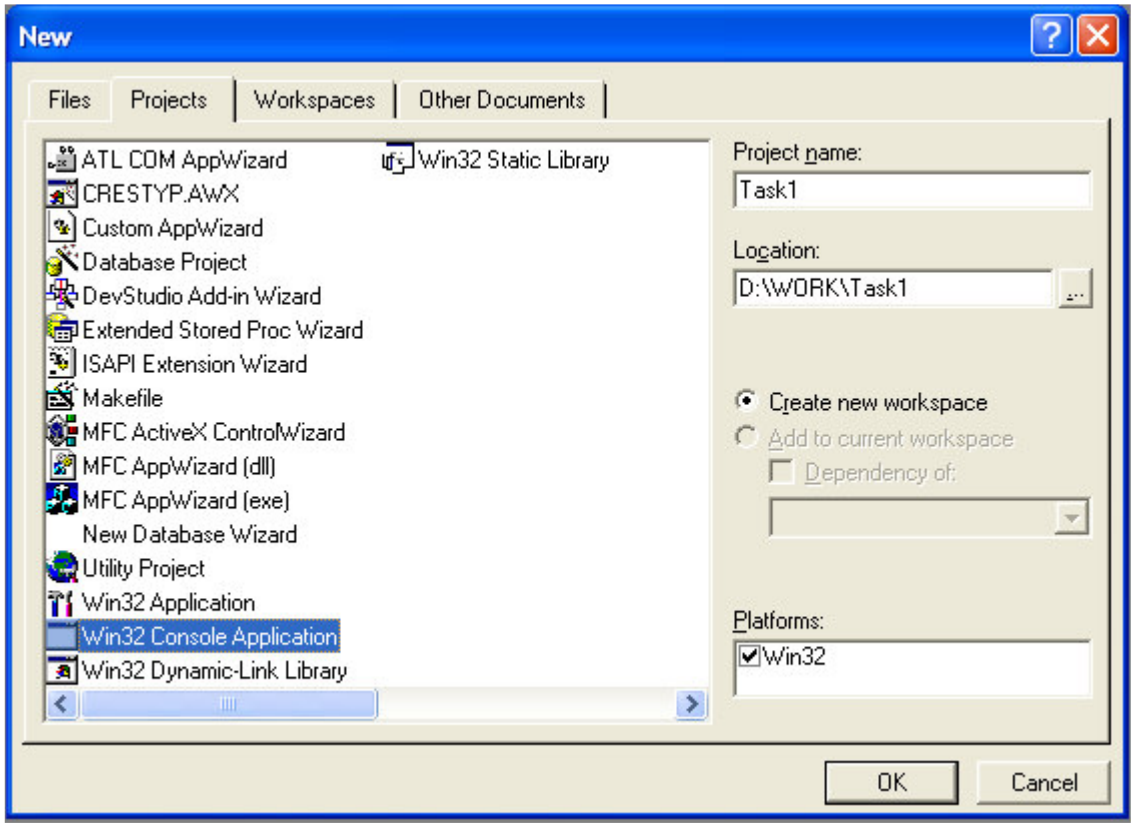



Рисунок 101. Выбор типа создаваемого приложения

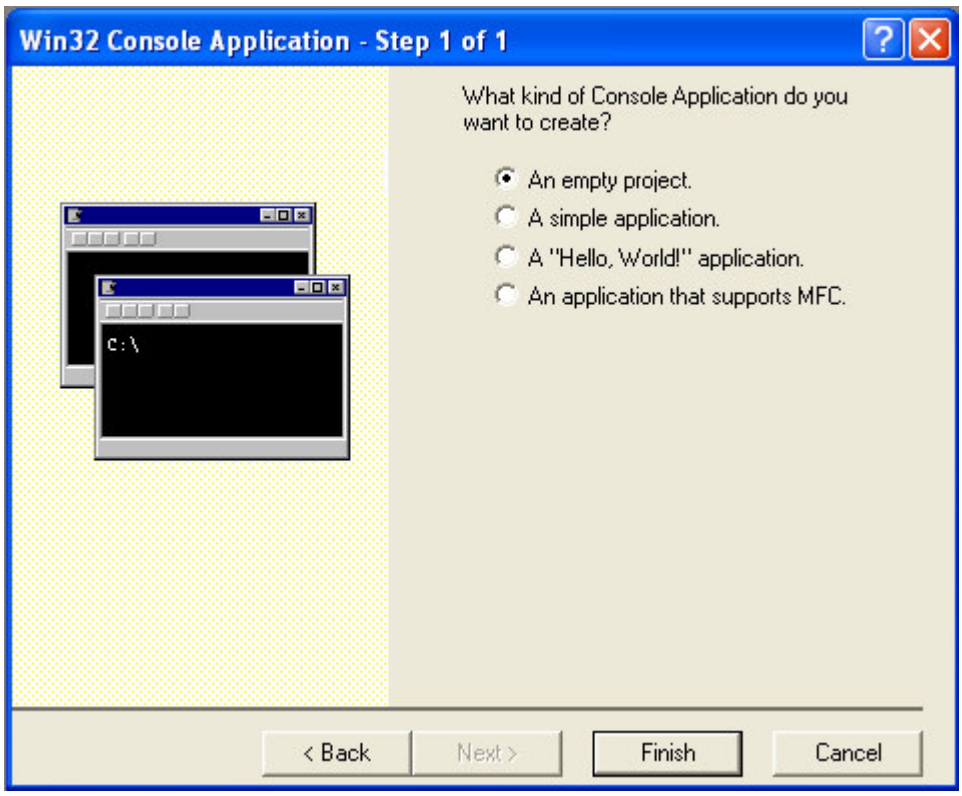



Рисунок 102. Генерация приложения

Появится окно ***New Project Information*** (информация о новом проекте) со спецификациями проекта и информацией о каталоге, в котором будет размещен создаваемый проект (рисунок 103).

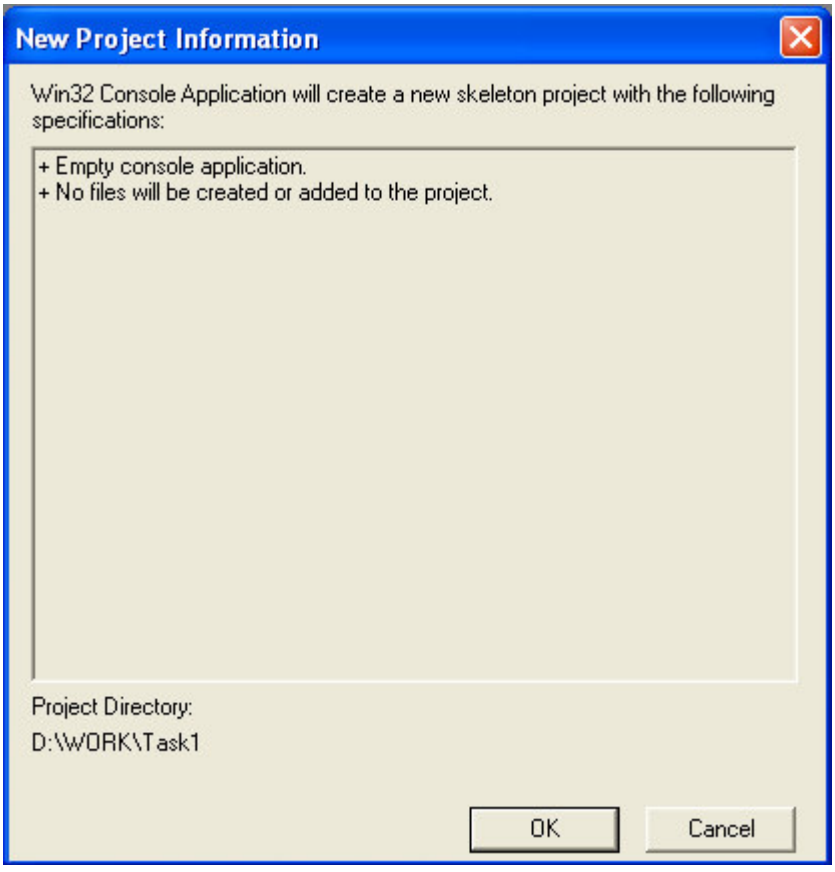


Рисунок 103. Завершение генерации приложения

7. Нажмите на кнопку **OK**. После выполненных шагов рабочий стол примет вид (рисунок 104).

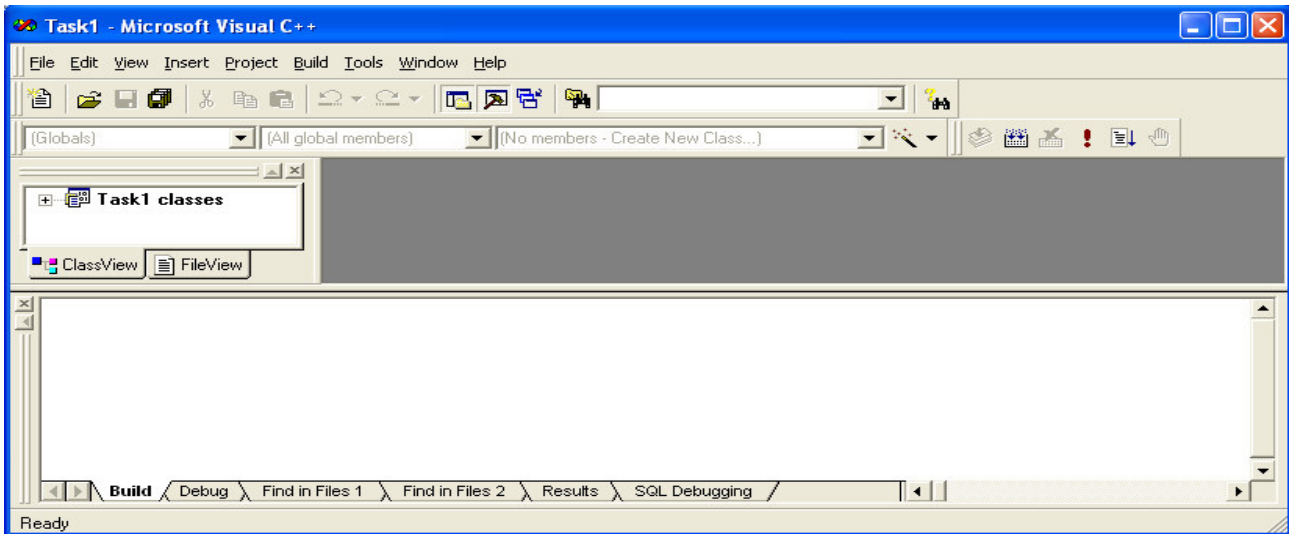



Рисунок 104. Вид главного окна Visual C++ после генерации приложения

8. В папке Task1, созданной мастером приложений, появятся файлы Task1.dsw, Task1.dsp, Task1.ncb и папка Debug (или Release – в зависимости от конфигурации проекта):

• Task1.dsw – файл рабочей области проекта, используемый внутри интегрированной среды разработки. Он объединяет всю информацию о проектах, входящих в данную рабочую область;

• Task1.dsp – проектный файл, используемый для построения (building*)* отдельного проекта или подпроекта;

• Task1.ncb – служебный файл. Он создается компилятором и содержит информацию, которая используется в инструменте интегрированной среды под названием ClassView.

Просмотрите вкладки ClassView *и* FileView окна Project Workspace.

При создании консольного приложения можно или добавить уже существующий файл с исходным кодом, который был создан при помощи других оболочек, или создать новый файл во встроенном текстовом редакторе среды программирования Visual Studio.

9. Выполните команду File – New… Выберите вкладку Files. Выберите тип файла C++ Source File. В поле File Name введите нужное имя файла Task1.cpp (желательно, чтобы имя файла с исходным кодом совпадало с именем проекта). Установите флажок Add to project (рисунок 105).

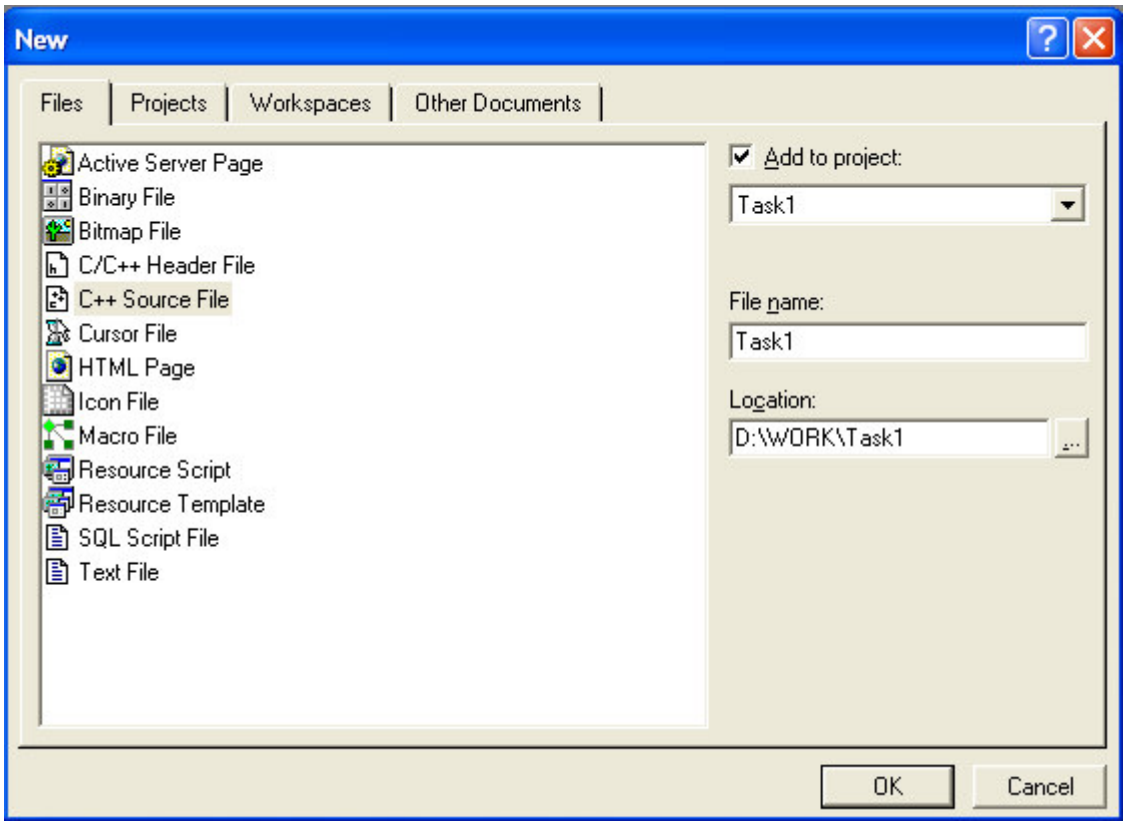



Рисунок 105. Добавление файлов в проект

10. После нажатия на кнопку OK в окне Project Workspace в папке Source Files списка файлов FileView проекта появится файл Task1.cpp, окно редактора Editor станет белым (рисунок 106).

11. Введите текст программы:

```
#include <iostream.h>
#include <math.h>
void main() {
    // Константы – ускорение свободного падения и число pi
    const double g=9.8;
    const double pi=3.1415;
    // Начальные и расчетные параметры задачи (скорость, угол и время полета)
    double v,alpha,T;
    // Момент времени и координаты
    double t,x,y;
// Ввод параметров
    cout<<"v = "; cin>>v;
    cout<<"alpha = "; cin>>alpha;
    alpha=alpha*pi/180;
    T=2*v*sin(alpha)/g;
    cout<<"t < "<<T<<": "; cin>>t;
    x=v*t*cos(alpha);
    y=v*t*sin(alpha)-g*t*t/2;
    cout<<"x = "<<x<<"\n";
    cout<<"y = "<<y<<"\n";
}
```

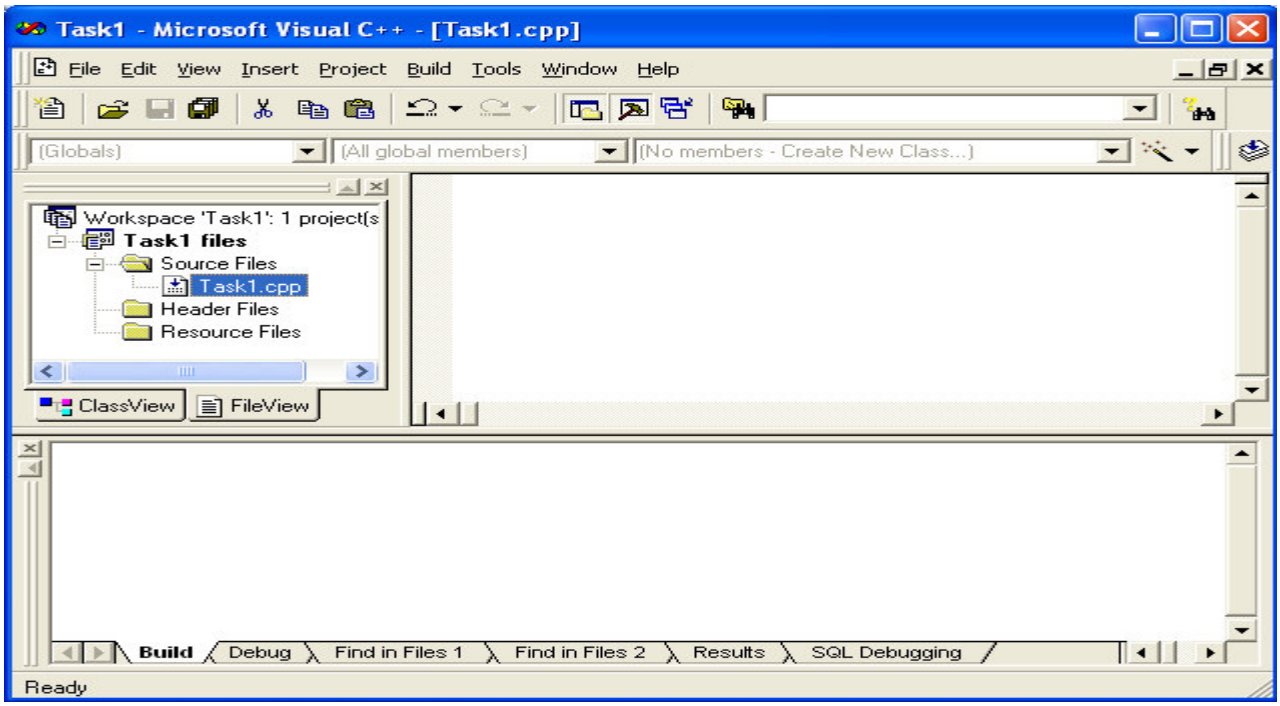



Рисунок 106. Редактирование кода

Первые две строки формируют раздел заголовков программы. Инструкция #include используется для присоединения внешнего файла. Внешние файлы подключаются для того, чтобы можно было использовать те или иные функции и утилиты. Файл iostream.h используется для поддержки системы ввода/вывода, файл math.h – для применения математических функций (таблица 14).

Таблица 14. Математические функции C++ (заголовочный файл math.h)

| Обращение | Тип аргумента | Тип результата | Функция |
|---|---|---|---|
| `abs(x)` | `int` | `int` | абсолютное значение целого числа |
| `acos(x)` | `double` | `double` | арккосинус (радианы) |
| `asin(x)` | `double` | `double` | арксинус (радианы) |
| `atan(x)` | `double` | `double` | арктангенс (радианы) |
| `ceil(x)` | `double` | `double` | ближайшее целое, не меньшее $x$ |
| `cos(x)` | `double` | `double` | косинус ($x$ в радианах) |
| `exp(x)` | `double` | `double` | $e^x$ — экспонента от $x$ |
| `fabs(x)` | `double` | `double` | абсолютное значение вещественного $x$ |
| `floor(x)` | `double` | `double` | наибольшее целое, не превышающее $x$ |
| `fmod(x,y)` | `double` `double` | `double` | остаток от деления нацело $x$ на $y$ |
| `log(x)` | `double` | `double` | логарифм натуральный — $\ln x$ |
| `log10(x)` | `double` | `double` | логарифм десятичный — $\lg x$ |
| `pow(x,y)` | `double` `double` | `double` | $x$ в степени $y$ — $x^y$ |
| `sin(x)` | `double` | `double` | синус ($x$ в радианах) |
| `sinh(x)` | `double` | `double` | гиперболический синус |
| `sqrt(x)` | `double` | `double` | корень квадратный (положительное значение) |
| `tan(x)` | `double` | `double` | тангенс ($x$ в радианах) |
| `tanh(x)` | `double` | `double` | гиперболический тангенс |

Вывод осуществляется посредством объекта cout из библиотеки iostream.h, ввод - посредством объекта cin из библиотеки iostream.h.

12. Сохраните проект с помощью команды File – Save Workspace (File – Save All).

13. Компиляцию, компоновку и выполнение проекта можно выполнить или через подменю Build главного окна, или при помощи кнопок панели инструментов, или при помощи комбинации горячих клавиш. Основные команды меню Build:

• Compile – компиляция выбранного файла. Результаты компиляции выводятся в окно Output;

• Build – компоновка проекта. Компилируются все файлы, в которых произошли изменения с момента последней компоновки. После компиляции происходит сборка *(link)* всех объектных модулей, включая библиотечные, в результирующий исполняемый файл. Сообщения об ошибках компоновки выводятся в окно ***Output***. Если обе фазы компоновки завершились без ошибок, среда программирования создаст исполняемый файл с расширением **.exe*, который можно запустить на выполнение;

• Rebuild All – то же, что и Build, но компилируются все файлы проекта независимо от того, были ли в них произведены изменения или нет;

• Execute – выполнение исполняемого файла, созданного в результате компоновки проекта. Если же в исходный текст были внесены изменения, то осуществляются перекомпилирование, перекомпоновка и выполнение.

Выполните команду Build – Execute Project1 или нажмите на кнопку  . Проверьте работу приложения (рисунки 107, 108).

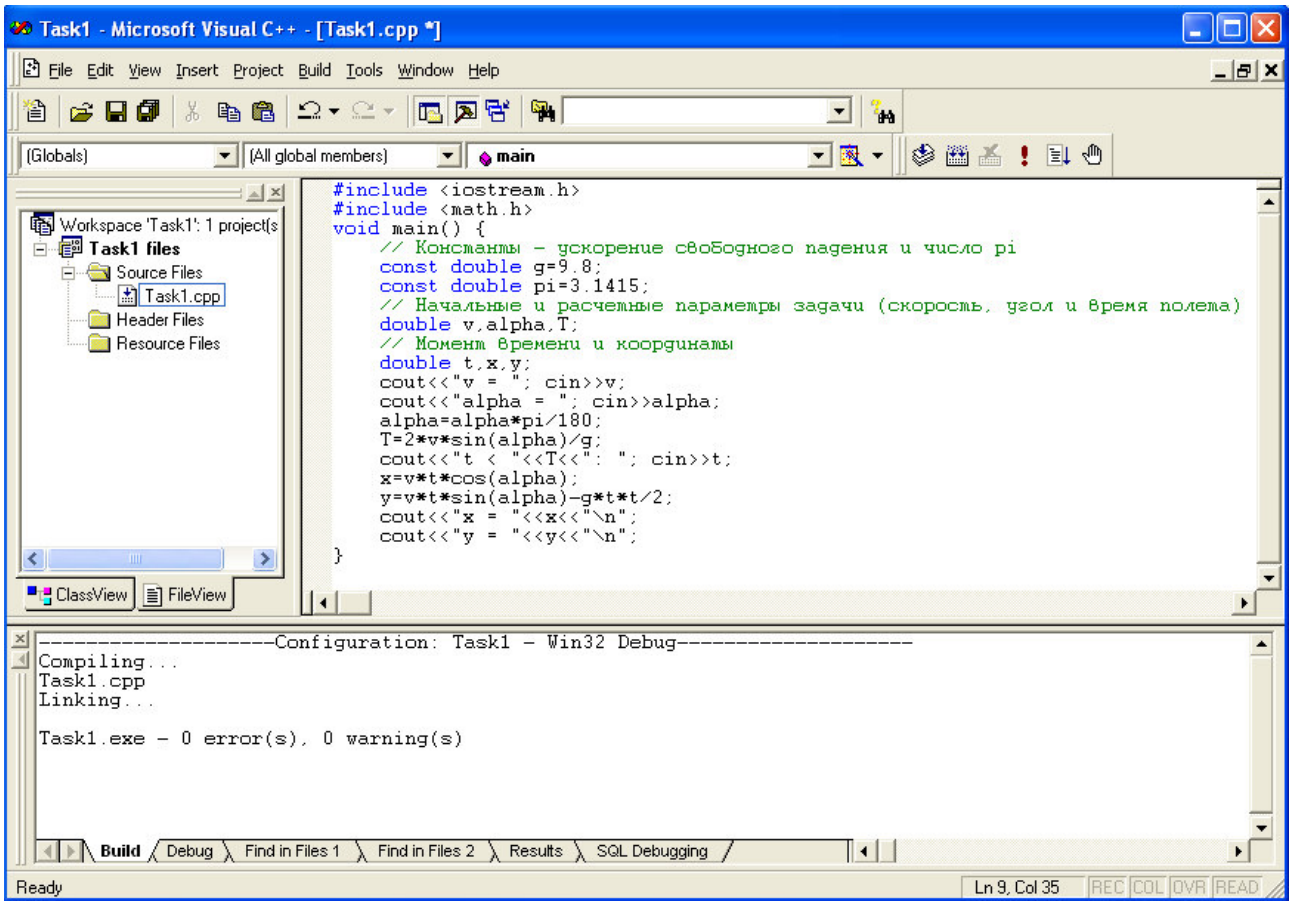



Рисунок 107. Запуск приложения

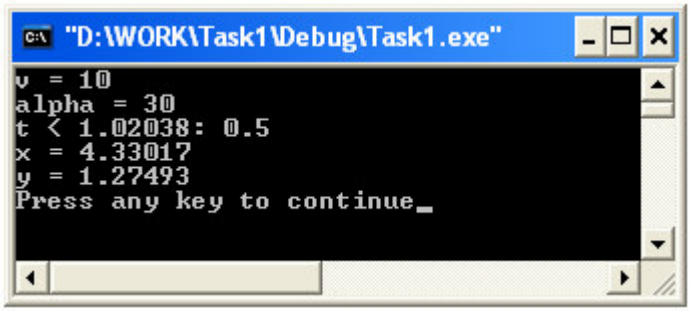



Рисунок 108. Результат работы приложения

14. Закройте проект с помощью команды **File – Close Workspace**.

*Задача 2*. Разработать приложение, решающее квадратное уравнение $ax^2 + bx + c = 0$.

Дискриминант квадратного уравнения $d = b^2 - 4ac$. По знаку дискриминанта можно определить, сколько корней имеет квадратное уравнение:

$d < 0$ – корней нет;

$d = 0$ – есть ровно один корень: $x = \frac{-b}{2a}$;

$d > 0$ – корней два: $x_1 = \frac{-b+\sqrt{d}}{2a}$, $x_2 = \frac{-b-\sqrt{d}}{2a}$.

1. Создайте новое консольное приложение Task2.

2. Введите текст программы:

```
#include <iostream.h>
#include <math.h>
#include "windows.h"          // включает процедуру CharToOem,
                              // используемую для вывода кирилицы
void main() {
    // Коэффициенты уравнения
    int a,b,c;
    // Дискриминант, корни уравнения
    double d,x,x1,x2;
    char sOutput[20];
    cout<<"a = "; cin>>a;
    cout<<"b = "; cin>>b;
    cout<<"c = "; cin>>c;
    d=b*b-4*a*c;
    if (d<0) {
            CharToOem("Корней нет", sOutput);
            cout<<sOutput<<"\n";
    } else
    if (d==0) {
            x=-b/(2*a);
            cout<<"x = "<<x<<"\n";
    }
    else {
            x1=(-b+sqrt(d))/(2*a);
            x2=(-b-sqrt(d))/(2*a);
            cout<<"x1 = "<<x1<<"\n";
            cout<<"x2 = "<<x2<<"\n";
    }
}
```

При вводе текста в редакторе Visual Studio используется кодовая страница 1251, а вывод текста в консольном приложении идет с применением кодовой страницы 866, поэтому при выводе кириллицы возникают проблемы. Для преобразования строки в нужную кодовую страницу можно использовать функцию CharToOem (windows.h).

3. Проверьте работу приложения.

| Проверяемый случай | Коэффициент | | | Результат |
|---|---|---|---|---|
| | a | b | c | |
| $d < 0$ | 1 | 1 | - | Два |

| | | | | |
|---|---|---|---|---|
| | | | 2 | корня: x1=1, x2=-2 |
| $d = 0$ | 1 | 2 | 1 | Один корень: x=-1 |
| $d > 0$ | 2 | 1 | 2 | Корней нет |

4. Сохраните и закройте проект.

*Задача 3.* Перевести числовую оценку знаний обучающегося в ее словесный эквивалент: 5 – «отлично», 4 – «хорошо», 3 – «удовлетворительно», 2 – «неудовлетворительно».

1. Создайте новое консольное приложение Task3.
2. Введите текст программы:

```
#include <iostream.h>
#include "windows.h"
void main()
{
    int ball;
    char sOutput[20];
    CharToOem("Введите оценку: ", sOutput);
    cout<<sOutput; cin>>ball;
    switch (ball)
    {
            case 2: CharToOem("Это неудовлетворительно", sOutput); break;
            case 3: CharToOem("Это удовлетворительно", sOutput); break;
            case 4: CharToOem("Это хорошо", sOutput); break;
            case 5: CharToOem("Это отлично", sOutput); break;
            default: CharToOem("Нет такой оценки", sOutput);
    }
    cout<<sOutput<<endl;
}
```

3. Сохраните проект. Проверьте работу приложения (рисунок 109).

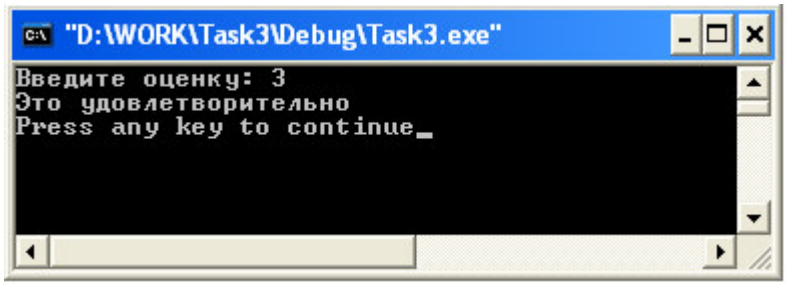



Рисунок 109. Результат работы приложения

4. Закройте проект.

*Задача 4.* Разработать приложение, вычисляющее синус заданного числа по формуле

$$\sin(x) \approx x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + \frac{(-1)^N x^{2N+1}}{(2N+1)!} = \sum_{n=0}^{N} \frac{(-1)^n x^{2n+1}}{(2n+1)!}.$$

1. Создайте новое консольное приложение Task4.
2. Введите текст программы:

```
#include <iostream.h>
// Граница ряда
const int N=100;
void main() {
    // Аргумент функции и рабочие переменные
    double x,q,s=0;
    // Индексная переменная
    int n;
    cout<<"x=";
    cin>>x;
    q=x;
    // Вычисление синуса
    for(n=1;n<=N;n++) {
            s+=q;
            q*=(-1)*x*x/(2*n)/(2*n+1); }
    // Вывод результата
    cout<<"sin("<<x<<")="<<s<<endl;
}
```

3. Сохраните проект. Проверьте работу приложения (рисунок 110).

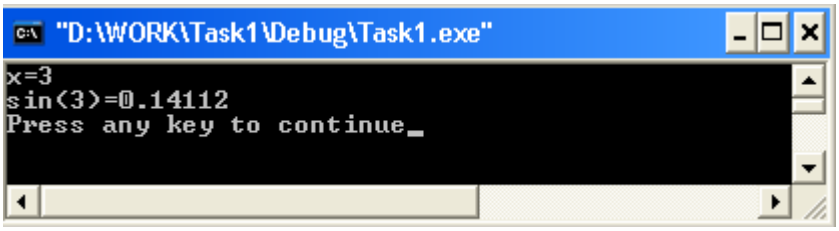



Рисунок 110. Результат работы приложения

4. Закройте проект.

*Задача 5.* Разработать приложение для вычисления произведения

$$\prod_{n=2}^{N} \left(1 - \frac{2}{n(n+1)}\right).$$

Верхняя граница произведения вводится пользователем с клавиатуры.

1. Создайте новое консольное приложение Task5.
2. Введите текст программы:

```
#include <iostream.h>
void main() {
    // Граница произведения
    int N;
    // Индексная переменная
    int n;
    // Переменная для записи произведения
```

```
    double s=1;
    // Ввод границы произведения
    cout<<"N=";
    cin>>N;
    // Вычисление произведения
    for(n=2;n<=N;n++) {
            s*=(1-(double)2/(n*(n+1))); }
    cout<<"s="<<s<<endl;
}
```

3. Сохраните проект. Проверьте работу приложения (рисунок 111).

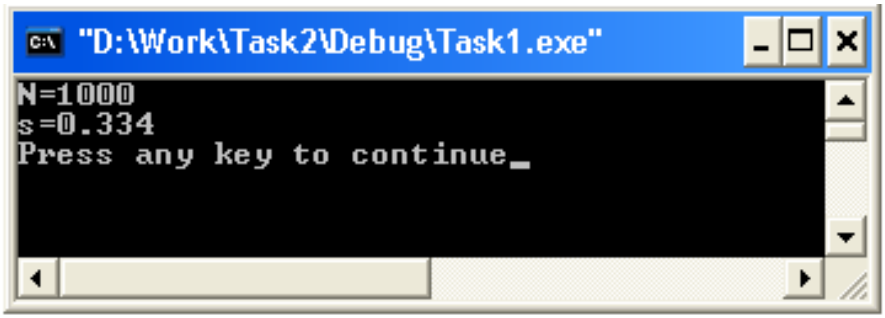



Рисунок 111. Результат работы приложения

4. Закройте проект.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу на языке C++, вычисляющую среднюю скорость движения мотоциклиста на участке от пункта А до В через пункт Б, если расстояние между пунктами А и Б составляет S1, а расстояние между пунктами Б и В равно S2. Время движения мотоциклиста между пунктами А и Б равно t1, а время движения между пунктами Б и В равно t2. Средняя скорость определяется как $V = (S_1 + S_2)/(t_1 + t_2)$. Параметры S1, S2, t1 и t2 вводятся пользователем с клавиатуры.

2. Составить программу на языке C++, определяющую, пройдет ли график функции $y = ax^2 + bx + c$ через заданную точку с координатами (m,n).

3. Составить программу на языке C++, вычисляющую косинус заданного числа по формуле

$$\cos(x) \approx 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + \frac{(-1)^N x^{2N}}{(2N)!} = \sum_{n=0}^{N} \frac{(-1)^n x^{2n}}{(2n)!}.$$

*Вариант 2*

1. Составить программу на языке C++, вычисляющую амплитуду колебаний А маятника. Маятник совершает колебания по закону $x(t) = A\sin(\omega t + \varphi_0)$. Частота колебаний $\omega$ известна. Известно также, что в начальный момент координата маятника положительна и в k раз меньше амплитуды А, а в момент времени t1 значение координаты маятника равно A1. Напишем программу, в которой определяется амплитуда колебаний А. Амплитуда колебаний определяется из соотношений $A/k = A\sin(\varphi_0)$ (что дает $\sin(\varphi_0) = 1/k$) и $A_1 = A\sin(\omega t_1 + \varphi_0)$. Можно найти точное аналитическое решение, но в данном случае в этом необходимости нет. Сначала по формуле $\varphi_0 = \arcsin(1/k)$ вычисляем начальную фазу $\varphi_0$, а затем по формуле $A = A_1/\sin(\omega t_1 + \varphi_0)$ вычисляем амплитуду А.

2. Составить программу на языке C++, которая возводит в квадрат те из трех заданных чисел, значения которых неотрицательны, и находит произведение полученных чисел.

3. Составить программу на языке C++, вычисляющую значение ряда ($|x| < 1$):

$$1 - 2x + 3x^2 - 4x^3 + \cdots + (-1)^N(N+1)x^N = \sum_{n=0}^{N}(-1)^n((n+1)x^n = \frac{1}{(1+x)^2}$$

*Вариант 3*

1. Составить программу на языке C++, которая введенное пользователем целое число умножает на 2 в целочисленной степени (показатель степени также вводится пользователем). Для выполнения умножения использовать оператор побитового сдвига.

2. Составить программу на языке C++, которая определяет, является ли среднее арифметическое заданных чисел целым числом.

3. Составить программу на языке C++, вычисляющую косинус заданного числа по формуле

$$\exp(x) \approx 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots + \frac{x^N}{(N)!} = \sum_{n=0}^{N}\frac{x^n}{(n)!}.$$

*Вариант 4*

1. Составить программу на языке C++, которая определяет высоту орбиты спутника h над поверхностью Земли, если известны масса M ≈ 5.96 x 1024 (кг) и радиус R ≈ 6.37 x 106 (м) Земли, масса спутника т, период его обращения T. Масса спутника в данном случае при расчете высоты орбиты не нужна, а период обращения вводится пользователем. При решении этой задачи необходимо воспользоваться тем, что сила гравитационного притяжения между Землей и спутником равна $F = G\frac{mM}{(R+h)^2}$, где G ≈ 6.672 x 1011 (Нм2/кг2) – универсальная гравитационная постоянная. С другой стороны, эту же силу по второму закону Ньютона можно записать как $F = ma$, где $a = \omega^2(R+h)$ есть центростремительное ускорение, а частота $\omega$ связана с периодом T соотношением $\omega = 2\pi/T$. Из этих соотношений получаем $\frac{4\pi^2 m(R+h)}{T^2} = GmM/(R+h)^2$, что дает $h = \sqrt[3]{\frac{GMT^2}{4\pi^2}} - R$.

2. Составить программу на языке C++, которая определяет, которая из точек находится ближе к началу координат. Расстояние между двумя точками определяется по формуле $r = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$.

3. Составить программу на языке C++, вычисляющую значение ряда ($|x| < 1$):

$$1 + 2x + 3x^2 + 4x^3 + \cdots + (N+1)x^N = \sum_{n=0}^{N}(n+1)x^n = \frac{1}{(1-x)^2}.$$

*Вариант 5*

1. Составить программу на языке C++, которая определяет длину окружности, площадь круга и объем шара одного и того же заданного радиуса. Длина окружности $L = 2\pi R$. Площадь круга $S = \pi R^2$. Объем шара $V = \frac{4}{3}\pi R^3$.

2. Составить программу на языке C++, которая вычисляет значение функции:

$$F(k) = \begin{cases} 9 - k, \text{если } k \geq 1.1 \\ \sin(3k)/(k^2+1), \text{если } k \leq 1.1 \end{cases}.$$

3. Составить программу на языке C++, вычисляющую значение ряда при заданном x по формуле

$$\frac{\sin(x)}{x} \approx 1 - \frac{x^2}{3!} + \frac{x^4}{5!} - \frac{x^6}{7!} + \cdots + \frac{(-1)^N x^{2N}}{(2N+1)!} = \sum_{n=0}^{N}\frac{(-1)^n x^{2n}}{(2n+1)!}.$$

**Лабораторный практикум № 13. Программирование задач с использованием одномерных и двухмерных массивов на языке С++**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 7.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение правил записи операторов цикла в языке С++, приемов программирования при работе с массивами.

***Вводная часть***

*Одномерные массивы*

Достаточно часто приходится иметь дело с наборами данных одного типа. Обычно такие данные в программе реализуют в виде массива. Под массивом подразумевают совокупность переменных одного типа, объединенных общим именем. Переменные, входящие в состав массива, называются элементами массива. Доступ к элементам массива осуществляется путем индексирования. Размерность массива определяется количеством индексов, необходимых для однозначного определения элемента массива. Массивы бывают статические (размер известен при компиляции программы) и динамические (размер определяется при выполнении программы). Рассмотрим статические одномерные массивы.

Под одномерным подразумевают массив, для индексации элементов которого используют один индекс. Как и в случае с обычной переменной, перед использованием массива его следует объявить. Объявление массива выполняется следующим образом: указывается тип данных, к которым принадлежат элементы массива, имя массива, а также его размер (количество элементов массива). Размер массива указывается в квадратных скобках сразу после имени массива. Например, командой int m[10] объявляется целочисленный массив с именем m, который состоит из 10 элементов. Размер массива задается числовым литералом или числовой константой. Размер массива должен быть известен на момент компиляции программы и не изменяется в процессе ее выполнения. Обращение к элементу массива выполняется через имя массива с индексом элемента в квадратных скобках. При этом следует помнить, что индексация элементов в С++ начинается с нуля. Таким образом, первым элементом означенного выше массива является m[0], а последним, десятым - элемент m[9]. Эта особенность языка С++ становится особенно важной с учетом того, что при компиляции и выполнении программ проверка на предмет выхода за пределы массива не выполняется.

В листинге приведен код программы, массив из 10 элементов которой заполняется случайными числами, а потом эти значения выводятся в строчку на экране.

```
#include <iostream.h>
void main() {
    int n[10];
    for(int i=0;i<10;i++) {
            n[i]=i;
            cout<<n[i]<<" ";
    }
    cout<<"\n";
}
```

В программе использован оператор цикла, индексная переменная i которого пробегает значения от 0 до 9 включительно. Значения от 0 до 9 присваиваются элементам массива и выводятся на экран командой cout<<n[i]<<" ".

Еще одна особенность С++ связана с тем, что имя массива (без индексов) является указателем на первый элемент массива. Например, если массив создается командой int n[10], то имя массива n является указателем (адресом) на первый элемент массива n[0]. В принципе, адрес этого элемента можно получить и стандартными методами, как для обычной переменной с помощью команды &n[0].

*Двумерные массивы*

Размерность массива может быть больше единицы (напомним, что размерность массива определяется количеством индексов, с помощью которых реализуется доступ к элементу массива). В таком случае говорят о многомерных массивах. Объявление многомерного массива выполняется так же просто, как и объявление одномерного массива, с той лишь разницей, что теперь для массива указывается размер по каждому из индексов. Для каждого индекса используется собственная пара квадратных скобок. При объявлении массива размер массива по соответствующему индексу также указывается в отдельных квадратных скобках. Среди многомерных массивов самым простым является двумерный массив. В известном смысле двумерный массив – это массив из одномерных массивов. Например, инструкцией double n[4][5] объявляется двумерный массив действительных чисел двойной точности размером 4x5. Как и ранее, чтобы обратиться к отдельному элементу массива, необходимо после имени массива указать индексы этого элемента (каждый индекс в отдельных квадратных скобках). Индексация по каждому индексу начинается с нуля. В листинге  приведен код программы, которой двумерный массив заполняется случайными числами с последующим выводом этих значений на экран:

```
#include <iostream.h>
void main() {
    int n[4][5];
    for(int i=0;i<4;i++) {
            for(int j=0;j<5;j++) {
                    n[i][j]=i+j;
                    cout<<n[i][j]<<" ";
            }
    cout<<"\n";
    }
}
```

Значения массива выводятся построчно в соответствии со структурой массива. Первый индекс массива определяет строку, второй индекс определяет столбец в этой строке. Так, элемент n[1][3] находится на пересечении второй строки и четвертого столбца. Результат выполнения программы будет следующим (рисунок 112).

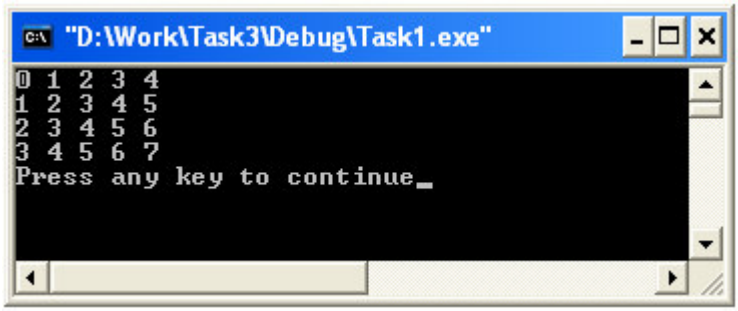



Рисунок 112. Результат работы приложения

Как и при работе с одномерными массивами, в двумерных массивах само по себе имя массива является ссылкой на первый элемент массива.

В C++ существует возможность инициализации массивов при их объявлении. Для инициализации одномерного массива после имени массива и размера через оператор присваивания указывается список со

значениями элементов. Список заключается в фигурные скобки, а сами значения разделяются запятыми. Синтаксис инициализации одномерного массива при объявлении имеет вид

тип имя_массива[размер]={значение!,значени2,...};

Например, инициализация двумерного массива может выглядеть так:

```
double numbers[3][2]        {1.1, 3.2,
                             8.3, 5.4,
                             9.5, 2.6};
```

Этот же массив с такими же начальными значениями мог быть объявлен в виде:

```
double numbers[3][2]        {{1.1, 3.2},
                             {8.3, 5.4},
                             {9.5, 2.6}};
```

В обоих случаях элемент массива numbers[0][0] получает значение 1.1, элемент numbers[0][1] получает значение 3.2, элемент numbers[1][0] получает значение 8.3 и т.д.

***Практическая часть***

*Задача 1.* Разработать приложение, осуществляющее поиск наименьшего элемента вектора. Элементы вектора вводятся пользователем с клавиатуры.

1. Создайте новое консольное приложение Task1.

2. Введите текст программы:

```
#include <iostream.h>
void main() {
    int i;
    double a[10];
    double res;
    for(i=0;i<10;i++) {
            cout<<"a["<<i+1<<"]="; cin>>a[i]; }
    res=a[0];
    for(i=1;i<10;i++) if (a[i]<res) res=a[i];
    cout<<"res="<<res<<endl;
}
```

3. Проверьте работу приложения (рисунок 113).

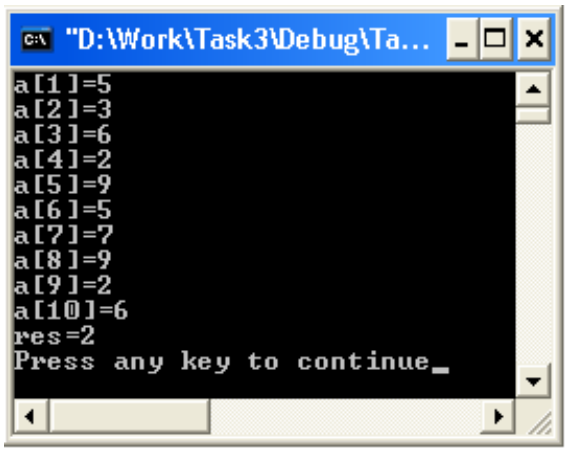



Рисунок 113. Результат работы приложения

4. Сохраните и закройте проект.

*Задача 2.* Написать программу, которая вычисляет среднее арифметическое значений элементов одномерного массива.

1. Создайте новое консольное приложение Task2.

2. Введите текст программы:

```
// Среднее значение массива
#include <iostream.h>
void main()
{
    const n=10;
    int i; double A[n], SA;
    for (i=0;i<n;i++) {cout<<"A["<<i<<"]="; cin>>A[i];}
    SA=0;
    for(i=0;i<n;i++) SA=SA+A[i];
    SA=SA/n;
    cout<<"\nSA="<<SA<<endl;
}
```

В этой программе обратите внимание на определение размера массива через константу.

3. Сохраните проект. Проверьте работу приложения (рисунок 114).

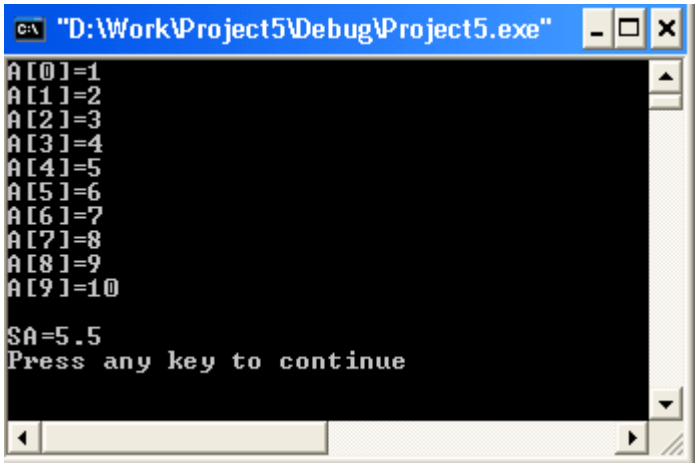



Рисунок 114. Результат работы приложения

4. Закройте проект.

*Задача 3.* Написать программу, которая сортирует одномерный массив «методом пузырька».

1. Создайте новое консольное приложение Task3.

2. Введите текст программы:

```
// Сортировка массива
#include <iostream.h>
void main()
{
    int X[]={6,4,9,3,2,1,5,7,8,10};
    int i,j,n,A;
    n=sizeof(X)/sizeof(X[0]);
    for (i=0;i<n-1;i++)
            for (j=0;j<n-1-i;j++)
                    if (X[j]>X[j+1]) {A=X[j]; X[j]=X[j+1]; X[j+1]=A;}
    for(i=0;i<n;i++) cout<<X[i]<<"  ";
}
```

В данной программе массив инициализирован. Его размер равен числу заданных значений. Чтобы сделать программу универсальной по отношению к размеру массива, значение размера вычисляется автоматически и заносится в переменную n. Для этого используется операция sizeof() – определение размера в байтах. Результат sizeof(X) равен размеру в памяти всего массива X – 20 байтам. Результат sizeof(X[0]) равен размеру одного элемента массива – 2 байтам. Отношение этих величин равно 10 – числу элементов массива.

3. Сохраните проект. Проверьте работу приложения (рисунок 115).

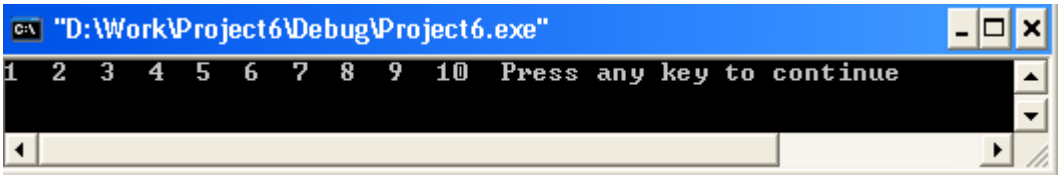



Рис. 115. Результат работы приложения

4. Закройте проект.

*Задача 4.* Написать программу, заполняющую матрицу случайными числами в диапазоне от 0 до 99 и осуществляющую поиск в ней максимального значения.

1. Создайте новое консольное приложение Task4.

2. Введите текст программы:

```
#include <iostream.h>
#include <iomanip.h>
#include <stdlib.h>
#define n 5
void main()
{
    int i, j, ImaxA, JmaxA, A[n][n];
    for (i=0;i<n;i++)
    {
            for (j=0;j<n;j++)
            {
                    A[i][j]=rand()%100;
                    cout<<setw(6)<<A[i][j];
            }
            cout<<endl;
    }
    ImaxA=JmaxA=0;
    for (i=0;i<n;i++)
    {
            for (j=0;j<n;j++)
                    if (A[i][j]>A[ImaxA][JmaxA]) { ImaxA=i; JmaxA=j; }
    }
    cout<<"Max: A["<<ImaxA<<"]["<<JmaxA<<"]="<<A[ImaxA][JmaxA]<<endl;
}
```

В данной программе имеются новые элементы, использование которых требует пояснения. В стандартной библиотеке с заголовочным файлом stdlib.h содержится функция, прототип которой имеет вид int rand(void).

Результатом этой функции является целое случайное число из диапазона от 0 до RAND_MAX. Значение константы RAND_MAX определено в заголовочном файле stdlib.h и обычно равно 32767 – максимально допустимому целому числу. Для получения случайных чисел в диапазоне от 0 до N-1 достаточно вычислить остаток от целого деления rand() на N.

Другим новым элементом в данной программе является использование манипуляторов для управления потоковым выводом с помощью стандартного объекта cout. Манипуляторы объявляются в заголовочном файле iomanip.h. Манипулятор setw(n) влияет на формат следующего элемента выходного потока. Он указывает на то, что значение будет выводиться в n позиций на экране (в программе n = 6).

3. Сохраните проект. Проверьте работу приложения (рисунок 116).

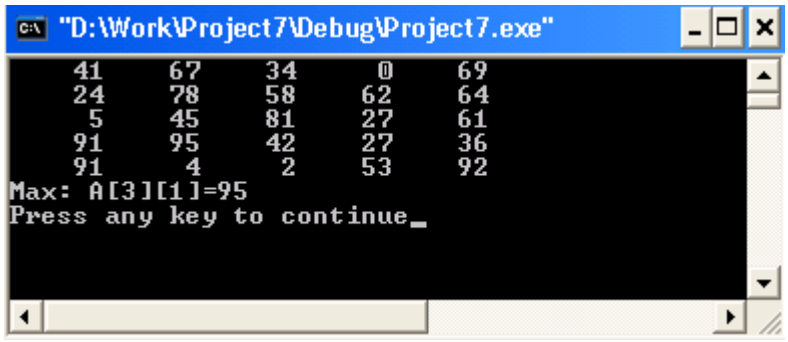



Рисунок 116. Результат работы приложения

4. Закройте проект.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу на языке C++, заполняющую одномерный массив A двадцатью первыми членами последовательности Фибоначчи (последовательности, в которой первые два члена равны 1, а каждый следующий равен сумме двух предыдущих: $a_1 = a_2 = 1; a_i = a_{i-1} + a_{i-2}$). Определить сумму элементов массива A и произведение первых семи элементов массива.

2. Составить программу на языке C++, заполняющую матрицу B размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $b_{i,j} = \sin(i^2) - \cos(j^2)$, где $i$ – номер строки, j – номер столбца. Определить количество отрицательных и положительных элементов матрицы B.

*Вариант 2*

1. Составить программу на языке C++, заполняющую одномерный массив A тридцатью первыми членами арифметической прогрессии (первый член прогрессии равен 10, ее разность равна 5): $a_1 = 10; a_{i+1} = a_i + 5$. Определить сумму элементов массива A.

2. Составить программу на языке C++, заполняющую матрицу B размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $b_{i,j} = \frac{1}{i+j}$, где i – номер строки, j – номер столбца. Определить количество элементов матрицы B, значение которых превышает заданное число.

*Вариант 3*

1. Составить программу на языке C++, заполняющую одномерный массив A: $a_i = \frac{c+d}{i}, i = \overline{1,20}, c = 20, d = 30$. Определить сумму элементов массива A.

2. Дано действительное число X=2.1. Составить программу на языке C++, формирующую действительную квадратную матрицу размером 4 на 4, элементы которой равны

$$B = \begin{pmatrix} X^1 & X^2 & X^3 & X^4 \\ X^5 & X^6 & X^7 & X^8 \\ X^9 & X^{10} & X^{11} & X^{12} \\ X^{13} & X^{14} & X^{15} & X^{16} \end{pmatrix}.$$

Определить сумму элементов матрицы B, лежащих выше и ниже главной диагонали.

*Вариант 4*

1. Составить программу на языке C++, заполняющую одномерный массив A пятнадцатью первыми членами геометрической прогрессии (первый член прогрессии равен 1, ее знаменатель равен 2): $a_1 = 1; a_{i+1} = a_i * 2$. Определить сумму элементов массива A.

2. Составить программу на языке C++, заполняющую матрицу B размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $b_{i,j} = \sin\frac{i^2 - j^2}{20}$, где i – номер строки, j – номер столбца. Определить количество отрицательных и положительных элементов матрицы B.

*Вариант 5*

1. Составить программу на языке C++, заполняющую одномерный массив A натуральными числами, делящимися нацело на 13 и лежащими в интервале [20;300]. Определить количество и сумму элементов одномерного массива A.

2. Составить программу на языке C++, формирующую квадратную матрицу B порядка n=20:

$$\begin{pmatrix} 1 & 2 & \ldots & n-1 & n \\ n+1 & n+2 & \ldots & 2n-1 & 2n \\ 2n+1 & 2n+2 & \ldots & 3n-1 & 3n \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ (n-1)n+1 & (n-1)n+2 & \ldots & n^2-1 & n^2 \end{pmatrix}.$$

Определить сумму и среднее арифметическое элементов матрицы B, лежащих на главной диагонали.

## Лабораторный практикум № 14. Работа со строковыми переменными на языке C++

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 7.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение приемов программирования при работе со строками в языке C++.

***Вводная часть***

*Массивы символов*

В C++ не существует встроенного типа для текстовых данных. Текстовые строки реализуются в виде массивов символов либо в виде объектов класса string.

По большому счету, массив символов мало чем отличается от массивов иных типов. Главная особенность связана с тем, что при объявлении массива из символов необходимо зарезервировать достаточно места для того, чтобы в такой массив можно было записывать строки разной длины. Другими словами, при реализации

строк в виде массивов существует принципиальное ограничение на длину строки. Такое ограничение существует и при использовании статических массивов других типов. Однако там эта проблема не столь актуальна. Поскольку при работе с символьными массивами речь идет о представлении типа данных, а не единичного объекта, необходимо предусмотреть возможность частого изменения данных. Принципиальным показателем при работе с текстовыми данными является длина строки. Поскольку в символьном массиве каждый символ строки соответствует элементу массива, длина строки напрямую имеет отношение к размеру символьного массива. С проблемой ограниченности размера строк связана еще одна проблема. Даже если размер массива достаточно велик для того, чтобы записывать в него строковые значения, необходим индикатор, который позволял бы определить, где в массиве записана полезная информация, а где начинается неинформативный «хвост». В качестве такого индикатора используют специальный символ '\0' , который называется нуль-символом.

Таким образом, строковая переменная реализуется в программе в виде массива символов. Признаком окончания строки является нуль-символ.

Чтобы вписать в массив строку, необходимо, чтобы размер массива, по крайней мере, на единицу превышал количество символов в строке. Этот дополнительный элемент необходим для записи нуль-символа '\0' окончания строки.

Объявляются массивы символов, как и прочие массивы: указывается тип элементов массива (для символьных массивов это char), название и размер массива. Пример объявления символьного массива:

```
char str[80] ;
```

В данном случае объявлен массив из 80 символов. При этом в такой массив можно записать строку с максимальной длиной в 79 символов. Инициализироваться символьные массивы могут так же, как и, например, числовые: после имени и размера массива указывается знак равенства и в фигурных скобках список символов, которые являются значениями элементов массива. Однако существует еще один более удобный способ инициализации символьного массива: вместо списка символов указывается в двойных кавычках текстовая строка. Далее в листинге приведен пример инициализации символьных массивов:

```
#include <iostream.h>
void main() {
    char str1[20]="hello";
    char str2[20]={'h','e','l','l','o','\0'};
    cout<<str1<<"\n";
    cout<<str2<<"\n";
}
```

Объявление символьных массивов с одновременной инициализацией выполняется командами char str1[20]="hello" и char str2[20]={'h','e','l','l','o','\0'}. В обоих случаях объявляется массив из 20 символов. Инициализация в первом случае выполняется путем указания текстового значения (значение заключено в двойные кавычки). Во втором случае это же значение передается в виде заключенного в фигурные скобки списка, причем в явном виде указывается нуль-символ окончания строки ' \0 ' . При инициализации массива с помощью текстового литерала нуль-символ добавляется автоматически.

Для вывода значений символьного массива на экран его имя указывается справа от оператора вывода (команды cout<<str1<<"\n" и cout<<str2<<"\n").

***Функции работы со строками***

Для функций обработки строк в программу необходимо включить заголовок <string.h>.

*Функция strcpy()*

Общий формат вызова функции strcpy():

```
strcpy(to, from);
```

Функция strcpy() копирует содержимое строки from в строку to. Массив, используемый для хранения строки to, должен быть достаточно большим, чтобы в него можно было поместить строку из массива from. В

противном случае массив to переполнится, т.е. произойдет выход за его границы, что может привести к прекращению работы программы.

*Функция strcat()*

Обращение к функции strcat() имеет следующий формат:

```
strcat(s1, s2);
```

Функция strcat() присоединяет строку s2 к концу строки s1, при этом строка s2 не изменяется. Обе строки должны завершаться нулевым символом. Результат вызова этой функции, т.е. результирующая строка s1, также будет завершаться нулевым символом. Программист должен позаботиться о том, чтобы строка s1 была достаточно большой и в нее поместилось, кроме ее исходного содержимого, содержимое строки s2.

*Функция strcmp()*

Обращение к функции strcmp() имеет следующий формат:

```
strcmp(s1, s2);
```

Функция strcmp() сравнивает строку s2 со строкой s1 и возвращает значение 0, если строки равны. Если строка s1 лексикографически (т.е. в соответствии с алфавитным порядком) больше строки s2, возвращается положительное число. Если строка s1 лексикографически меньше строки s2, возвращается отрицательное число.

При использовании функции strcmp() важно помнить, что она возвращает число 0 (т.е. значение false), если сравниваемые строки равны. Следовательно, если необходимо выполнить определенные действия при условии совпадения строк, надо использовать оператор НЕ (!). Например, условие, управляющее следующей if-инструкцией, даст истинный результат, если строка str содержит значение "C++":

```
if(!strcmp(str, "C++")) cout<<"Строка str содержит C++";
```

*Функция strlen()*

Общий формат вызова функции strlen():

```
strlen(s);
```

Здесь s – строка. Функция strlen () возвращает длину строки, указанной аргументом s.

Пример программы, использующей функции обработки строк:

```
#include <iostream.h>
#include <string.h>
void main()
{
    char s1[80], s2[80];
    strcpy(s1, "C++");
    strcpy(s2, " - это мощный язык.");
    cout<<"Длины строк: "<<strlen(s1)<<"  "<<strlen(s2)<<endl;
    if(!strcmp(s1,s2)) cout<<"Эти строки равны."<<endl;
    else cout<<"Эти строки не равны."<<endl;
    strcat(s1,s2);
    cout<<s1<<endl;
    strcpy(s2,s1);
    cout<<s1<<" и "<<s2<<endl;
    if(!strcmp(s1,s2))
            cout<<"Строки s1 и s2 теперь одинаковы."<<endl;
    }
```

Результат выполнения программы:

```
    Длины строк: 3  19
    Эти строки не равны.
    C++ - это мощный язык.
    C++ - это мощный язык. и C++ - это мощный язык.
```

Строки s1 и s2 теперь одинаковы.

***Практическая часть***

*Задача.* Ввести символьную строку. Перевернуть (обратить) эту строку. Например, если ввели строку «abcdef», то в результате должны получить «fedcba».

1. Создайте новое консольное приложение Task1.

2. Введите текст программы:

```
#include <iostream.h>
#include <string.h>
void main()
{
   char C,S[10];
   int i;
   cout<<"S=";cin>>S;
   for(i=0; i<=(strlen(S)-1)/2; i++)
   {
            C=S[i];
            S[i]=S[strlen(S)-i-1];
            S[strlen(S)-i-1]=C;
   }
   cout<<"S="<<S<<endl;
}
```

3. Проверьте работу приложения (рисунок 117).

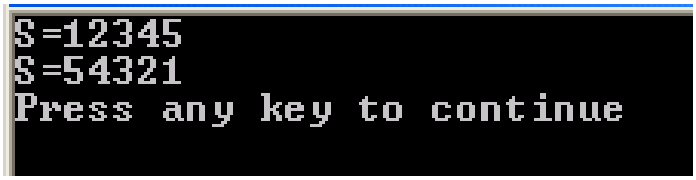



Рисунок 117. Результат работы приложения

4. Сохраните и закройте проект.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и удаляет пробелы в начале строки.

2. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и определяет количество символов * в строке.

*Вариант 2*

1. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и проверяет, является ли строка целым числом без знака.

2. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и заменяет точки в строке восклицательными знаками.

*Вариант 3*

1. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и проверяет, является ли строка двоичным числом без знака.

2. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и заменяет двоеточия в строке точками с запятой.

*Вариант 4*

1. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и проверяет, является ли строка восьмеричным числом без знака.

2. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и определяет количество символов, предшествующих двоеточию.

*Вариант 5*

1. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и проверяет, является ли строка шестнадцатеричным числом без знака.

2. Составить программу на языке C++, которая выполняет ввод строки с клавиатуры и определяет количество символов, предшествующих точке.

## Лабораторный практикум № 15. Программирование задач с использованием функций на языке C++

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 7.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение основных принципов работы с указателями и функциями в языке программирования C++.

***Вводная часть***

*Указатели*

Указатель – это адрес поля памяти, занимаемого программным объектом.

Пусть в программе определены три переменные разных типов:

```
int a=5;
char c='G';
float r=1.2E8;
```

Эти величины размещаются в памяти компьютера следующим образом:

| Память | FFC0 | FFC1 | FFC2 | FFC3 | FFC4 | FFC5 | FFC6 |
|---|---|---|---|---|---|---|---|
| Переменные | a | | c | r | | | |
| Значения | 5 | | 'G' | 1.2*108 | | | |

Операция & – адрес. Применение этой операции к имени переменной дает в результате ее адрес в памяти. Для переменных из данного выше примера: &a равно FFC0, &c – FFC2, &r – FFC3.

*Описание указателей.* Для хранения адресов используются переменные типа «указатель». Формат описания таких переменных следующий:

```
тип *имя_переменной;
```

Примеры описания указателей:

```
int *pti;
char *ptc;
float *ptf;
```

После такого описания переменная pti может принимать значение указателя на величину целого типа; переменная ptc предназначена для хранения указателя на величину типа char; переменная ptf – на величину типа float.

Указателям могут присваиваться значения адресов объектов только того типа, с которым они описаны. В нашем примере допустимы операторы:

```
pti=&a;
ptc=&c;
ptf=&r;
```

В результате указатели примут следующие значения:

pti – FFC0, ptc – FFC2, ptf – FFC2.

Как и для других типов данных, значения указателей могут инициализироваться при описании. Например:

int a=5; int *pti=&a;

char c='G'; char *ptc=&c;

float r=1.28E8; float *ptf=&r;

В заголовочном stdio.h определена константа – нулевой указатель с именем NULL. Ее значение можно присваивать указателю. Например:

Ptf=NULL;

Не надо думать, что после этого указатель ptf будет ссылаться на нулевой байт памяти. Нулевой указатель обозначает отсутствие конкретного адреса ссылки.

Использованный в описаниях указателей символ * (звездочка) в данном контексте является знаком операции разадресации. С ее помощью можно сослаться через указатель на соответствующую переменную.

После приведенных выше описаний в записи выражений этой программы взаимозаменяемыми становятся a и *pti, c и *ptc, r и *ptf. Например, два оператора

x=a+2; и x=*pti+2;

тождественны друг другу. В результате выполнения оператора

cout<<*pti<<a;

на экран выведется 55.

*Операции над указателями.* Записывая выражения и операторы, изменяющие значения указателей, необходимо помнить главное правило: единицей изменения значения указателя является размер соответствующего ему типа.

Продемонстрируем это правило на определенных выше указателях. Выполнение операторов

pti=pti+1; или pti++;

изменит значение указателя pti на 2, в результате чего он примет значение FFC2. В результате выполнения оператора pti--; значение указателя уменьшится на 2 и станет равным FFBE.

Аналогично для указателей других типов:

• ptc++; увеличит значение указателя на 1;

• ptf++; увеличит значение указателя на 4.

*Указатели и массивы.* Имя массива трактуется как указатель-константа на массив.

Пусть, например, в программе объявлен массив

int: X[10] ;

В таком случае X является указателем на нулевой элемент массива в памяти компьютера. В связи с этим истинным является отношение

X= =&X[0].

Отсюда следует, что для доступа к элементам массива кроме индексированных имен можно использовать разадресованные указатели по принципу:

имя [индекс] тождественно * (имя + индекс)

Например, для описанного выше массива X взаимозаменяемы следующие обозначения элементов:

X[5] или *(X+5) или *(5+X)

Для указателей работают свои правила сложения. Поскольку X – указатель на величину целого типа, то X+5 увеличивает значение адреса на 10.

В языке C++ символ [ играет роль знака операции сложения адреса массива с индексом элемента массива.

Из сказанного должно быть понятно, почему индекс первого элемента массива всегда нуль. Его адрес должен совпадать с адресом массива:

X[0]= = * (X+0).

Поскольку имя массива является указателем-константой, то его нельзя изменять в программе, т.е. ему нельзя ничего присваивать. Например, если описаны два одинаковых по структуре массива

int X[10], Y[10] ;

то оператор присваивания X=Y будет ошибочным. Такое возможно в Паскале, но недопустимо в C++. Пересылать значения одного массива в другой можно только поэлементно.

Теперь рассмотрим двумерные массивы. Пусть в программе присутствует описание

int P[5][10];

Это матрица из пяти строк и десяти чисел в каждой строке. Двумерный массив расположен в памяти в последовательности по строкам. По-прежнему P является указателем-константой на массив, т.е. на элемент P[0][0]. Индексированное имя P[i] обозначает i-ю строку. Ему тождественно следующее обозначение в форме разадресованного указателя:

*(P+i*10)

Обращение к элементу массива P[2][4] можно заменить на *(P+2*10+4). В общем случае эквивалентны обозначения:

P[i][j] и * (P+i*10+j)

Здесь дважды работает операция «квадратная скобка». Последнее выражение можно записать иначе, без явного указания на длину строки матрицы p:

*(*(P+i)+j)

Очевидно, что по индукции для ссылки на элемент трехмерного массива A[i][j][k] справедливо выражение

*(*(*(A+i)+j)+k) и т.д.

*Функции*

*Основы использования функций.* Функция – это подпрограмма, которая содержит одну или несколько инструкций и выполняет определенную задачу.

C++ функции имеют следующий формат:

```
тип_возвращаемого_значения имя (список_параметров)
{
   // тело функции
}
```

Вложенность описаний функций запрещается. Если функция не возвращает никакого значения, необходимо указывать тип void. Если функция возвращает значение, оно должно иметь тип, совместимый с указанным в определении функции. Функция завершается при достижении закрывающейся фигурной скобки или инструкции return.

Пример:

```
#include <iostream.h>
void myfunc();                          // Прототип функции myfunc()
void main()
{
   cout<<"---main_1---"<<endl;
   myfunc();                            // Вызов функции myfunc()
   cout<<"---main_2---"<<endl;
}
void myfunc()                                   // Определение функции myfunc()
{
   cout<<"---myfunc---"<<endl;
}
```

*Параметры функции.* Функции можно передать одно или несколько значений. При этом необходимо объявить переменные, которые получат передаваемые значения. Эти переменные называются параметрами функции.

Пример функции, вычисляющей объем параллелепипеда:

```
#include <iostream.h>
void box(int length, int width, int height);
void main()
{
   box(7, 20, 4);
   box(50, 3, 2);
```

```
    box(8, 6, 9);
}
void box(int length, int width, int height)
{
    cout<<"V="<<length * width * height<<endl;
}
```

*Использование инструкции return.* Иногда требуется более гибкое средство управления возвратом из функции. Таким средством служит инструкция return. Инструкция return имеет две формы применения: первая позволяет возвращать значение (return 0;), а вторая – нет (return;). При обнаружении инструкции return управление программой немедленно передается инициатору ее вызова. Любой код функции, расположенный за инструкцией return, игнорируется. Функция может содержать несколько инструкций return:

```
void f()
{
    // …
switch(c) {
    case 'a': return;
    case 'b': // …
    case 'c': return;
}
    if (count < 100) return;
    // …
}
```

*Возврат значений из функции.* Функция может возвращать значение инициатору своего вызова. Для этого используется инструкция

return значение;

Эту форму инструкции return можно использовать только с не void-функциями. Функция может возвращать данные любого допустимого в C++ типа, за исключением массива.

Пример:

```
#include <iostream.h>
int box(int length, int width, int height);
void main()
{
    int answer;
    answer = box(10,11,3);
    cout<<"V="<<answer<<endl;
}
int box(int length, int width, int height)
{
    return length * width * height;
}
```

*Использование указателей для передачи параметров функции.* В языке C++ возможна только односторонняя передача значений фактических параметров из вызывающей программы к формальным параметрам вызываемой функции. Возвращаемое значение несет сама функция, используемая в качестве операнда в выражении. Отсюда, казалось бы, следует неукоснительное правило: в процессе выполнения функции не могут изменяться значения переменных в вызывающей программе. Однако это правило можно обойти, если в качестве параметров функции использовать указатели.

В следующем примере функция swap() производит обмен значениями двух переменных величин, заданных своими указателями в аргументах.

```
void swap(int *a, int *b)
```

```
{
    int c;
    c=*a; *a=*b; *b=c;
}
```

Если в основной программе имеется следующий фрагмент:

```
void main()
{
    int x=1, y=2;
    swap(&x, &y);
    printf("x=%d  y=%d", x, y);
}
```

то на экран будет выведено

x=2  y=1

т.е. переменные x и y поменялись значениями.

Все выглядит очень похоже на то, как если бы в Паскале использовали процедуру обмена с var-параметрами. И тем не менее передача параметров здесь тоже происходит по значению, только этими значениями являются указатели. После обращения к функции указатель *a* получил адрес переменной x, указатель b – адрес переменной y. После этого переменная x в основной программе и разадресованный указатель *a в функции оказываются связанными с одной ячейкой памяти; так же – y и *b.

Таким образом, можно сделать вывод о том, что использование указателей в параметрах функции позволяет моделировать работу процедур.

*Рекурсивные функции.* Функция, которая вызывает саму себя, называют рекурсивной. Классическим примером рекурсии является вычисление факториала числа.

Пример:

```
#include <iostream.h>
int fact(int n);
void main()
{
    cout<<"n!="<<fact(4)<<endl;
}
int fact(int n)
{
    int answer;
    if (n==1) return 1;
    answer = fact(n-1) * n;
    return answer;
}
```

***Практическая часть***

*Задача 1.* Написать программу, которая с помощью функции вычисляет максимальное из двух целых чисел, полученных в качестве аргументов.

1. Создайте новое консольное приложение Task1.

2. Введите текст программы:

```
#include <iostream.h>
int max(int a, int b);
void main()
{
    int x,y;
    cout<<"x="; cin>>x;
    cout<<"y="; cin>>y;
    cout<<"max="<<max(x,y)<<endl;
```

```
}
int max(int a, int b)
{
    if (a>b) return a;
    else return b;
}
```

3. Проверьте работу приложения (рисунок 118).

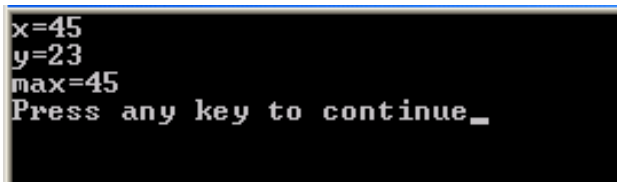



Рисунок 118. Результат работы приложения

4. Сохраните и закройте проект.

*Задача 2*. Написать программу, которая с помощью рекурсивной функции вычисляет $x^n$:

$$x^n = \begin{cases} 1, \text{если } n = 0 \\ x^{n-1} \cdot x, \text{если } n \neq 0 \end{cases}.$$

1. Создайте новое консольное приложение Task2.
2. Введите текст программы:

```
#include <iostream.h>
int st(int x, int n);
void main()
{
    int x,n;
    cout<<"x="; cin>>x;
    cout<<"n="; cin>>n;
    cout << "answer=" << st(x,n) <<endl;
}
int st(int x, int n)
{
    if (n==0)
            return 1;
    else
            return st(x,n-1) * x;
}
```

3. Проверьте работу приложения (рисунок 119).

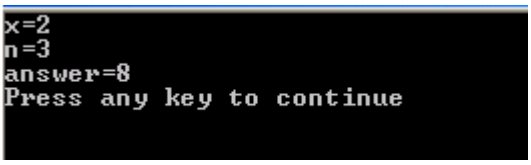



Рисунок 119. Результат работы приложения

4. Сохраните и закройте проект.

*Задача 3*. Составим программу решения следующей задачи. Дана вещественная матрица A[M][N]. Требуется вычислить и вывести евклидовы нормы строк этой матрицы.

Евклидовой нормой вектора называют корень квадратный из суммы квадратов его элементов:

$$L = \sqrt{\sum_{i=1}^{n} x_i^2} \ .$$

Если строку матрицы рассматривать как вектор, то данную формулу надо применить к каждой строке. В результате необходимо получить M чисел.

1. Создайте новое консольное приложение Task3.

2. Введите текст программы:

```
#include <iostream.h>
#include <math.h>
double Norma(int n, double X[]);
void main()
{
    double A[2][3];
    int i,j;                ;
    for (i=0;i<2;i++)
    {
            for (j=0;j<3;j++)
            {
                    cout<<"A["<<i<<","<<j<<"]="; cin>>A[i][j];
            }
    }
    for (i=0;i<2;i++)
            cout<<"Norma "<<i<<"="<<Norma(3,A[i])<<endl;
}
double Norma(int n, double X[])
{
    int i;
    double S=0;
    for (i=0; i<n; i++) S+=X[i]*X[i];
    return sqrt(S);
}
```

3. Проверьте работу приложения (рисунок 120).

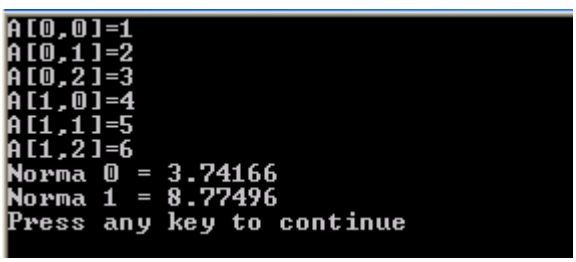



Рисунок 120. Результат работы приложения

4. Сохраните и закройте проект.

***Самостоятельная работа***

*Вариант 1*

1. Составить программу на языке C++, которая с помощью функции определяет среди чисел 1+1/2, 1+1/2+1/3, 1+1/2+1/3+1/4,… первое, большее *a*.

2. Составить программу на языке C++, которая с помощью рекурсивной функции вычисляет заданный член числового ряда:

$$a_n = 2 + \frac{1}{a_{n-1}},\ a_1 = 2.$$

3. Составить программу на языке C++, заполняющую матрицу A размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $a_{i,j} = \frac{1}{i+j}$, где i – номер строки, j – номер столбца. С помощью подпрограммы-функции вычислить среднее арифметическое каждой из строк матрицы A.

*Вариант 2*

1. Составить программу на языке C++, которая с помощью функции определяет первое значение функции z = xk / k, большее *a*, если k = 1, 2, 3, ….

2. Составить программу на языке C++, которая с помощью рекурсивной функции вычисляет заданный член числового ряда:

$$a_n = \frac{1}{2}\operatorname{tg} a_{n-1},\quad a_1 = 0{,}5.$$

3. Составить программу на языке C++, заполняющую матрицу A размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $a_{i,j} = \sin\frac{i^2 - j^2}{20}$, где i – номер строки, j – номер столбца. С помощью подпрограммы-функции вычислить среднее арифметическое каждой из строк матрицы A.

*Вариант 3*

1. Составить программу на языке C++, которая с помощью функции определяет первое значение функции 1/x+1/x2+1/x3+1/x4 + …, большее *a*.

2. Составить программу на языке C++, которая с помощью рекурсивной функции вычисляет заданный член числового ряда:

$$a_n = \frac{2 + a_{n-1}^2}{2a_{n-1}},\ a_1 = 2.$$

3. Составить программу на языке C++, заполняющую матрицу A размером 20 строк и 30 столбцов, каждый элемент которой получается по формуле $a_{i,j} = \sin(i^2) - Cos(j^2)$, где i – номер строки, j – номер столбца. С помощью подпрограммы-функции вычислить сумму элементов каждой из строк матрицы A.

*Вариант 4*

1. Составить программу на языке C++, которая с помощью функции определяет первое значение функции $z = \frac{1}{(k+1)^2}$, меньшее *a*, если k = 1, 2, 3, ….

2. Составить программу на языке C++, которая с помощью рекурсивной функции вычисляет заданный член числового ряда:

$$a_n = \frac{a_{n-1} + a_{n-2}}{2},\ a_1 = 1,\ a_2 = 2.$$

3. Составить программу на языке C++, формирующую квадратную матрицу A порядка n=20:

$$\begin{pmatrix} 1 & 2 & \ldots & n-1 & n \\ n+1 & n+2 & \ldots & 2n-1 & 2n \\ 2n+1 & 2n+2 & \ldots & 3n-1 & 3n \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ (n-1)n+1 & (n-1)n+2 & \ldots & n^2-1 & n^2 \end{pmatrix}.$$

С помощью подпрограммы-функции вычислить сумму элементов каждой из строк матрицы A.

*Вариант 5*

1. Составить программу на языке C++, которая с помощью функции определяет первое значение функции $z = \frac{1}{(2k)^2}$, меньшее *a*, если k = 1, 2, 3, ….

2. Составить программу на языке C++, которая с помощью рекурсивной функции вычисляет заданный член числового ряда:

$$a_n = e^{-a_{n-1}},\ a_1 = 0.$$

3. Дано действительное число X=2.1. Составить программу на языке C++, формирующую действительную квадратную матрицу размером 4 на 4, элементы которой равны:

$$\begin{pmatrix} X^1 & X^2 & X^3 & X^4 \\ X^5 & X^6 & X^7 & X^8 \\ X^9 & X^{10} & X^{11} & X^{12} \\ X^{13} & X^{14} & X^{15} & X^{16} \end{pmatrix}.$$

С помощью подпрограммы-функции вычислить сумму элементов каждой из строк матрицы A.

## Лабораторный практикум № 16. Разработка приложения, основанного на диалоге на языке C++

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 8.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение основных принципов работы с библиотекой классов MFC.

***Вводная часть***

*Библиотека MFC*

Библиотека классов MFC (Microsoft Foundation Classes) – базовый набор (библиотека) классов, написанных на языке C++ и предназначенных для упрощения и ускорения процесса программирования для Windows. Библиотека содержит многоуровневую иерархию классов, насчитывающую около 200 членов. Они дают возможность создавать Windows-приложения на базе объектно-ориентированного подхода. С точки зрения программиста, MFC представляет собой каркас, на основе которого можно писать программы для Windows.

Библиотека MFC разрабатывалась для упрощения задач, стоящих перед программистом. Как известно, традиционный метод программирования под Windows требует написания достаточно длинных и сложных программ, имеющих ряд специфических особенностей. В частности, для создания только каркаса программы таким методом понадобится около 75 строк кода. По мере же увеличения сложности программы ее код может

достигать поистине невероятных размеров. Однако та же самая программа, написанная с использованием MFC, будет примерно в три раза меньше, поскольку большинство частных деталей скрыто от программиста.

Одним из основных преимуществ работы с MFC является возможность многократного использования одного и того же кода. Так как библиотека содержит много элементов, общих для всех Windows-приложений, нет необходимости каждый раз писать их заново. Вместо этого их можно просто наследовать. Кроме того, интерфейс, обеспечиваемый библиотекой, практически независим от конкретных деталей, его реализующих. Поэтому программы, написанные на основе MFC, могут быть легко адаптированы к новым версиям Windows (в отличие от большинства программ, написанных обычными методами).

Еще одним существенным преимуществом MFC является упрощение взаимодействия с прикладным программным интерфейсом (API) Windows. Библиотека MFC объединяет (путем инкапсуляции) функции API в логически организованное множество классов.

Библиотека классов MFC содержит большое количество разнообразных классов. Каждый класс, как правило, содержит от нескольких единиц до нескольких десятков различных методов и элементов данных.

Рассмотрим кратко назначение некоторых основных классов библиотеки MFC и их связь друг с другом.

*Класс CObject*

Подавляющее большинство классов MFC наследовано от базового класса CObject, лежащего в основе всей иерархии классов этой библиотеки. Методы и элементы данных класса CObject представляют наиболее общие свойства наследованных из него классов MFC.

Класс CObject, а также все классы, наследованные от него, обеспечивают возможность сохранения объектов класса в файлах на диске с их последующим восстановлением.

Для объектов классов, наследованных от базового класса CObject, уже во время работы приложения можно получить разнообразную информацию о классе объекта.

Ряд методов класса CObject предназначен для получения дампа объектов класса во время отладки приложения. Эта особенность класса может ускорить процесс поиска ошибок в приложении.

*Класс CCmdTarget*

Непосредственно от класса CObject наследуются ряд классов, которые сами являются базовыми для остальных классов MFC. В первую очередь это класс CCmdTarget, представляющий основу структуры любого приложения. Основной особенностью класса CCmdTarget и классов, наследованных от него, является то, что объекты этих классов могут получать от операционной системы сообщения и обрабатывать их.

*Классы CWinThread и CWinApp*

От класса CCmdTarget наследуется класс CWinThread, представляющий подзадачи приложения. Простые приложения имеют только одну подзадачу. Эта подзадача, называемая главной, представляется классом CWinApp, наследованным от класса CWinThread.

*Класс CDocument*

Большинство приложений работают с данными или документами, хранимыми на диске в отдельных файлах. Класс CDocument, наследованный от базового класса CCmdTarget, служит для представления документов приложения.

*Классы CDocTemplate, CSingleDocTemplate и CMultiDocTemplate*

Еще один важный класс, наследуемый от CCmdTarget, называется CDocTemplate. От этого класса наследуется два класса: CSingleDocTemplate и CMultiDocTemplate. Все эти классы предназначены для синхронизации и управления основными объектами, представляющими приложение, – окнами, документами и используемыми ими ресурсами.

*Класс CWnd*

Практически все приложения имеют пользовательский интерфейс, построенный на основе окон. Это может быть диалоговая панель, одно окно или несколько окон, связанных вместе. Основные свойства окон представлены классом CWnd, наследованным от класса CCmdTarget.

Программисты очень редко создают объекты класса CWnd. Класс CWnd сам является базовым классом для большого количества классов, представляющих разнообразные окна.

*Класс CFrameWnd*

Класс CFrameWnd представляет окна, выступающие в роли обрамляющих окон. К ним относятся главные окна приложений. От этого класса также наследуются классы CMDIChildWnd и CMDIFrameWnd, используемые для отображения окон многооконного интерфейса MDI. Класс CMDIFrameWnd представляет главное окно приложения MDI, а класс CMDIChildWnd - дочерние окна MDI. Класс CMiniFrameWnd применяется для отображения окон уменьшенного размера. Такие окна обычно используются для отображения в них панели управления.

*Классы элементов управления*

Для работы с элементами управления (кнопками, полосами прокрутки, редакторами текста и т.д.) в библиотеке MFC предусмотрены специальные классы, наследованные непосредственно от класса CWnd. Перечислим эти классы:

CAnimate – используется для отображения видеоинформации;

CBitmapButton – кнопка с рисунком;

CButton – кнопка;

CComboBox – список с окном редактирования;

CEdit – поле редактирования;

CHeaderCtrl – заголовок для таблицы;

CHotKeyCtrl – предназначен для ввода комбинации клавиш акселераторов;

CListBox – список;

CListCtrl – может использоваться для отображения списка пиктограмм;

CProgressCtrl – линейный индикатор;

CPropertySheet – блокнот, может состоять из нескольких страниц;

CRichEditControl – окно редактирования, в котором можно редактировать форматированный текст;

CScrollBar – полоса просмотра;

CSliderCtrl – движок;

CSpinButtonCtrl – обычно используется для увеличения или уменьшения значения какого-либо параметра;

CStatic – статический орган управления;

CTabCtrl – набор "закладок";

CToolBarCtrl – панель управления;

CToolTipCtrl – маленькое окно, содержащее строку текста;

CTreeCtrl – орган управления, который позволяет просматривать иерархические структуры данных.

*Классы CControlBar, CToolBar, CStatusBar, CDialogBar*

Класс CControlBar и классы, наследуемые от него, предназначены для создания управляющих панелей. Такие панели могут содержать различные органы управления и отображаются, как правило, в верхней или нижней части главного окна приложения.

Так, класс CToolBar предназначен для создания панели управления. Эта панель обычно содержит ряд кнопок, дублирующих действие меню приложения.

Класс CStatusBar управляет панелью состояния. Панель состояния отображается в виде полосы в нижней части окна. В ней приложение может отображать всевозможную информацию, например, краткую подсказку о выбранной строке меню.

Большие возможности представляет управляющая панель, созданная на основе класса CDialogBar. Такая панель использует обычный шаблон диалоговой панели, которую можно разработать в редакторе ресурсов Visual Studio.

*Класс CView и классы, наследованные от него*

Большой интерес представляют класс CView и классы, наследуемые от него. Эти классы представляют окно просмотра документов приложения. Именно окно просмотра используется для вывода на экран документа, с которым работает приложение. Через это окно пользователь может изменять документ.

Разрабатывая приложение, программисты наследуют собственные классы просмотра документов либо от базового класса CView, либо от одного из нескольких порожденных классов, определенных в библиотеке MFC.

Классы, наследованные от CCtrlView, используют для отображения готовые органы управления. Например, класс CEditView использует орган управления edit (редактор).

Класс CScrollView представляет окно просмотра, которое имеет полосы свертки. В классе определены специальные методы, управляющие полосами просмотра.

Класс CFormView позволяет создать окно просмотра документа, основанное на диалоговой панели. От этого класса наследуется еще два класса: CRecordView и CDaoRecordView. Эти классы используются для просмотра записей баз данных.

*Класс CDialog и классы, наследованные от него*

От базового класса наследуются классы, управляющие диалоговыми панелями. Если необходимо создать диалоговую панель, можно наследовать класс от CDialog.

Вместе с диалоговыми панелями обычно используется класс CDataExchange. Класс CDataExchange обеспечивает работу процедур обмена данными DDX (Dialog Data Exchange) и проверки данных DDV (Dialog Data Validation), используемых для диалоговых панелей. В отличие от CDialog класс CDataExchange не наследуется от какого-либо другого класса.

От класса CDialog наследуется ряд классов, представляющих собой стандартные диалоговые панели для выбора шрифта, цвета, вывода документа на печать, поиска в документе определенной последовательности символов, а также поиска и замены одной последовательности символов другой последовательностью.

Чтобы создать стандартный диалог, можно просто определить объект соответствующего класса. Дальнейшее управление такой панелью осуществляется методами класса.

*Класс CDC*

Для отображения информации в окне или на любом другом устройстве приложение должно получать так называемый контекст отображения. Основные свойства контекста отображения определены в классе CDC. От него наследуется 4 различных класса, представляющие контекст отображения различных устройств.

Дадим краткое описание классов, наследованных от CDC.

CClientDC – контекст отображения, связанный с внутренней областью окна (client area). Для получения контекста конструктор класса вызывает функцию программного интерфейса GetDC, а деструктор – функцию ReleaseDC.

CMetaFileDC – класс предназначен для работы с метафайлами.

CPaintDC – конструктор класса CPaintDC для получения контекста отображения вызывает метод CWnd::BeginPaint, деструктор – метод CWnd::EndPaint. Объекты данного класса можно использовать только при обработке сообщения WM_PAINT. Это сообщение обычно обрабатывает метод OnPaint.

CWindowDC – контекст отображения, связанный со всем окном. Для получения контекста конструктор класса вызывает функцию программного интерфейса GetWindowDC, а деструктор – функцию ReleaseDC.

*Класс CGdiObject*

Для отображения информации используются различные объекты графического интерфейса – GDI-объекты. Для каждого из этих объектов библиотека MFC содержит описывающий его класс, наследованный от базового класса CGdiObject.

Для работы с GDI-объектами используются классы:

CBitmap – растровое изображение bitmap;

CBrush – кисть;

CFont – шрифт;

CPalette – палитра цветов;

CPen – перо;

CRgn – область внутри окна.

*Класс CMenu*

Практически каждое приложение имеет собственное меню. Оно, как правило, отображается в верхней части главного окна приложения. Для управления меню в состав MFC включен специальный класс CMenu, наследованный непосредственно от базового класса CObject.

Для управления меню и панелями используется также класс CCmdUI. Этот класс не наследуется от базового класса CObject.

Объекты класса CCmdUI создаются, когда пользователь выбирает строку меню или нажимает кнопки панели управления. Методы класса CCmdUI позволяют управлять строками меню и кнопками панели управления. Например, существует метод, который делает строку меню неактивной.

*Другие классы*

В MFC включено несколько классов, обеспечивающих поддержку приложений, работающих с базами данных. Это такие классы, как CDataBase, CRecordSet, CDaoDataBase. CDaoRecordSet, CDaoQueryDef, CDaoTableDef, CDaoWorkSpace, CLongBinary, CFieldExchange и CDaoFieldExchange.

Библиотека MFC позволяет создавать многозадачные приложения. Для синхронизации отдельных задач приложения предусмотрен ряд специальных классов. Все они наследуются от класса CSyncObject, представляющий собой абстрактный класс.

В некоторых случаях требуется, чтобы участок программного кода мог выполняться только одной задачей. Такой участок называют критической секцией кода. Для создания и управления критическими секциями предназначены объекты класса CCriticalSection.

Объекты класса CEvent представляют событие. При помощи событий одна задача приложения может передать сообщение другой.

Объекты класса CMutex позволяют в данный момент предоставить ресурс в пользование одной только задаче. Остальным задачам доступ к ресурсу запрещается.

Объекты класса CSemafore представляют собой семафоры. Семафоры позволяют ограничить количество задач, которые имеют доступ к какому-либо ресурсу.

Для программистов, занимающихся сетевыми коммуникациями, в состав библиотеки MFC включены классы CAsyncSocket и наследованный от него класс CSocket. Эти классы облегчают задачу программирования сетевых приложений.

Кроме уже описанных классов библиотека MFC включает большое количество классов, предназначенных для организации технологии OLE.

***Использование средств разработки***

К средствам разработки относятся MFC AppWisard, ClassWizard и редактор ресурсов.

Благодаря MFC AppWizard среда разработчика позволяет быстро создавать шаблоны новых приложений. При этом программисту не приходится писать ни одной строчки кода. Достаточно ответить на ряд вопросов, касающихся того, какое приложение требуется создать, и исходные тексты шаблона приложения вместе с файлами ресурсов готовы. Эти тексты можно оттранслировать и получить готовый загрузочный модуль приложения.

Конечно, никакие средства автоматизированной разработки не смогут создать программу полностью без участия программиста. Прикладную часть приложения придется разрабатывать самостоятельно.

Для создания ресурсов приложения предназначен редактор ресурсов. Он позволяет быстро создавать новые меню, диалоговые панели, добавлять кнопки к панели управления ToolBar и т.д.

Средство ClassWizard позволяет подключить к созданным и отредактированным ресурсам управляющий ими код. Большую часть работы по описанию и определению функций, обрабатывающих сообщения от меню, органов управления диалоговых панелей и т.д., также берет на себя средство ClassWizard.

Библиотека MFC содержит большое количество классов, структур, констант и т.д. Для того чтобы текст MFC-приложений был более легким для понимания, принято применять ряд соглашений для используемых имен и комментариев.

Названия всех классов и шаблонов классов библиотеки MFC начинаются с заглавной буквы C. При наследовании классов от классов MFC можно давать им любые имена. Рекомендуется начинать их названия с заглавной буквы C. Это сделает исходный текст приложения более ясным для понимания.

Чтобы отличить элементы данных, входящих в класс, от простых переменных, их имена принято начинать с префикса m_. Названия методов классов, как правило, специально не выделяются, но обычно их начинают с заглавной буквы.

Библиотека MFC включает в себя, помимо классов, набор служебных функций. Названия этих функций начинаются с символов Afx, например AfxGetApp. Символы Afx являются сокращением от словосочетания Application FrameworkX, означающих основу приложения, его внутреннее устройство.

Символы Afx встречаются не только в названии функций MFC. Многие константы, макрокоманды и другие символы начинаются с этих символов. В общем случае Afx является признаком, по которому можно определить принадлежность того или иного объекта (функции, переменной, ключевого слова или символа) к библиотеке MFC.

Когда приложение разрабатывается средствами MFC AppWizard и ClassWizard, они размещают в исходном тексте приложения комментарии следующего вида:

//{{AFX_ ... //}}AFX_

Такие комментарии образуют блок кода программы, который управляется только средствами MFC AppWizard и ClassWizard. Пользователь не должен вручную вносить изменения в этом блоке. Для этого необходимо употреблять средства ClassWizard.

В тех местах, где можно вставить свой код, MFC AppWizard и ClassWizard, как правило, помещают комментарии:

//TODO:

***Работа со стандартными элементами управления***

*Методы класса CButton – интерфейс для кнопок (BUTTON), флажков (CHECKBOX), переключателей (RADIOBUTTON)*

HBITMAP GetBitmap() const;

Возвращает дескриптор растрового изображения, сопоставленного кнопке. Если такового не существует, то возвращается NULL.

HBITMAP SetBitmap(HBITMAP hBitmap);

Сопоставляет кнопке растровое изображение. Значением параметра должен быть дескриптор растрового изображения. Правила размещения растрового изображения такие же, как и у значка.

UINT GetState() const;

Возвращает описание набора текущих состояний кнопки. Чтобы выделить из этого описания значения конкретных типов состояния, можно использовать маски:

0x0003 – выделяет собственное состояние кнопки. Применимо только к флажку или переключателю. Если результат побитового умножения дает 0, значит, кнопка находится в невыбранном состоянии, 1 – в выбранном, 2 – в неопределенном.

0x0004 – выделяет состояние первого типа. Ненулевой вариант означает, что кнопка "нажата", нулевой – кнопка свободна.

0x0008 – выделяет положение фокуса. Ненулевой вариант – кнопка в фокусе клавиатуры.

int GetCheck() const;

Возвращает собственное состояние флажка или переключателя. Возвращаемое значение может принимать одно из значений: 0 – кнопка не выбрана; 1 – кнопка выбрана; 2 – кнопка в неопределенном состоянии. Если кнопка не является ни переключателем, ни флажком, возвращается 0.

void SetCheck(int nCheck);

Устанавливает собственное состояние флажка или переключателя. Значения задаются из набора: 0 – невыбранное; 1 – выбранное; 2 – неопределенное.

UINT GetButtonStyle() const;

Возвращает стиль кнопки.

void SetButtonStyle(UINT nStyle, BOOL bRedraw=TRUE);

Устанавливает стиль кнопки. Если параметр bRedraw равен TRUE, кнопка перерисовывается.

HICON GetIcon() const;

Возвращает дескриптор пиктограммы, сопоставленной кнопке. Если у кнопки нет сопоставленной пиктограммы, возвращает NULL.

HICON SetIcon(HICON hIcon);

Сопоставляет кнопке пиктограмму. Значением параметра при вызове должен быть дескриптор пиктограммы.

Пиктограмма автоматически помещается на поверхность кнопки и сдвигается в ее центр. Если поверхность кнопки меньше пиктограммы, она обрезается со всех сторон до размеров кнопки. Положение пиктограммы может быть выровнено и не по центру. Для этого нужно, чтобы кнопка имела одно из следующих свойств: BS_LEFT, BS_RIGHT, BS_CENTER, BS_TOP, BS_BOTTOM, BS_VCENTER

Данный метод устанавливает для кнопки только одну пиктограмму, которая будет наравне с текстом присутствовать при любом ее состоянии.

*Методы класса CEdit – окно простейшего редактора текста*

DWORD GetSel() const;

void GetSel(int& nStartChar, int& nEndChar) const;

Получает первую и последнюю позиции выделенного текста. Для значения типа DWORD младшее слово содержит позицию первого символа, старшее – последнего символа.

void SetSel(DWORD dwSelection, BOOL bNoScroll=FALSE);

void SetSel(int nStartChar, int nEndChar, BOOL bNoScroll=FALSE);

Устанавливает новое выделение текста, задавая первый и последний выделенный символ. Значение FALSE параметра bNoScroll должно отключать перемещение курсора в область видимости.

void ReplaceSel(LPCTSTR lpszNewText);

Заменяет выделенный текст на строку, передаваемую в параметре lpszNewText.

void Clear();

Удаляет выделенный текст.

void Copy();

Копирует выделенный текст в буфер.

void Cut();

Переносит (копирует и удаляет) выделенный текст в буфер обмена.

void Paste();

Вставляет текст из буфера обмена, начиная с позиции, в которой находится курсор.

BOOL Undo();

Отмена последней операции, выполненной редактором. Если редактор однострочный, возвращается всегда неотрицательное значение, иначе неотрицательное значение возвращается лишь в случае успешной замены.

BOOL CanUndo() const;

Определяет, можно ли отменить последнюю операцию редактора.

void EmptyUndoBuffer();

Сбрасывает флаг undo, сигнализирующий о возможности отмены последней операции редактора, и тем самым делает невозможным отмену. Этот флаг сбрасывается автоматически при выполнении методов SetWindowText и SetHandle.

BOOL GetModify() const;

Возвращает неотрицательное значение, если содержимое окна редактирования не модифицировалось. Информация о модификации поддерживается в специальном флаге, обнуляемом при создании окна редактирования и при вызове метода:

void SetModify(BOOL bModified=TRUE);

Устанавливает или сбрасывает флаг модификации. Флаг сбрасывается при вызове метода с параметром FALSE и устанавливается при модификации содержимого окна редактирования или при вызове SetModify с параметром TRUE.

BOOL SetReadOnly(BOOL bReadOnly=TRUE);

Устанавливает режим просмотра (bReadOnly=TRUE) или редактирования (bReadOnly=FALSE).

TCHAR GetPasswordChar() const;

Возвращает символ, который при выводе пароля будет появляться на экране вместо символов, набираемых пользователем. Если такой символ не определен, возвращается 0. Устанавливается этот символ методом (по умолчанию используется "*"):

void SetPasswordChar(TCHAR ch);

void LimitText(int nChars=0);

Устанавливает максимальную длину в байтах текста, который может ввести пользователь. Если значение параметра равно 0, длина текста устанавливается равной UINT_MAX.

*Методы работы с многострочным редактором*

void LineScroll(int nLines, int nChars=0);

Прокручивает текст в области редактирования. Параметр nLines задает число строк для вертикальной прокрутки. Окно редактирования не прокручивает текст дальше последней строки. При положительном значении параметра область редактирования сдвигается вдоль текста к последней строке, при отрицательной – к первой.

Параметр nChars задает число символов для горизонтальной прокрутки. Окно редактирования прокручивает текст вправо, даже если строки закончились. В этом случае в области редактирования появляются пробелы. При положительном значении параметра область редактирования сдвигается вдоль к концу строки, при отрицательном - к началу.

int GetFirstVisibleLine() const;

Возвращает номер первой видимой строки.

int GetLineCount() const;

Возвращает число строк текста, находящегося в буфере редактирования. Если текст не вводился, возвращает 1.

int GetLine(int nIndex, LPTSTR lpszBuffer) const;

int GetLine(int nIndex, LPTSTR lpszBuffer, int nMaxLength) const;

Копирует строку с номером, равным значению параметра nIndex, в буфер, заданный параметром lpszBuffer. Первое слово в буфере должно задавать его размер. При вызове второго варианта метода значение параметра nMaxLength копируется в первое слово буфера. Метод возвращает число в действительности скопированных байтов. Если номер строки больше или равен числу строк в буфере окна редактирования, возвращает 0. Текст копируется без каких-либо изменений, нуль-символ не добавляется.

int LineIndex(int nLine=-1) const;

Возвращает номер первого символа в строке. Неотрицательное значение параметра принимается в качестве номера строки. Значение –1 задает текущую строку. Если номер строки больше или равен числу строк в буфере окна редактирования (строки нумеруются с 0), возвращается 0.

*Методы класса CListBox – список элементов, которые пользователь может просматривать и выбирать*

void ResetContent();

Очищает содержимое списка, делая его пустым.

int AddString( LPCSTR lpszItem);

Добавляет строку lpszItem в список и сортирует его, если при создании включено свойство Sort. В противном случае элемент добавляется в конец списка.

int DeleteString( UINT nIndex);

Удаляет из списка элемент с индексом nIndex. Индексация элементов начинается с 0.

int GetCurSel() const;

Получает индекс элемента, выбранного пользователем.

int SetCurSel( int nSelect);

Отмечает элемент с индексом nSelect как выбранный элемент списка. Если значение параметра равно -1, список не будет содержать отмеченных элементов.

int GetText( int nIndex, LPSTR lpszBuffer) const;

void GetText( int nIndex, CString& rString) const;

Копирует элемент с индексом nIndex в буфер.

int SetTopIndex( int nIndex);

Организует прокрутку списка в окне так, чтобы элемент с индексом nIndex был видимым.

int FindString( int nStartAfter, LPCSTR lpszItem) const;

Организует поиск в списке и возвращает в качестве результата индекс элемента списка, префикс которого совпадает со строкой lpszItem. Результат не зависит от регистра, в котором набирались символы сравниваемых строк. Параметр nStartAfter задает начало поиска, но поиск идет по всему списку. Он начинается от элемента, следующего за nStartAfter, до конца списка и затем продолжается от начала списка до элемента с индексом nStartAfter. В качестве результата выдается первый найденный элемент, удовлетворяющий условиям поиска. Если такого нет, результат получает значение LB_ERR.

int FindStringExact( int nIndexStart, LPCSTR lpszFind) const;

Этот метод отличается от предыдущего тем, что теперь не префикс элемента должен совпадать со строкой lpszFind, а сам элемент. Поиск по-прежнему не чувствителен к регистру, в котором набираются символы.

*Методы класса CComboBox – элемент управления, объединяющий поле редактора текста со списком*

int GetCurSel() const;

Возвращает целочисленный указатель выбранной строчки.

int SetCurSel(int nSelect);

Ставит указатель на строчку с номером nSelect.

int GetLBText(int nIndex, LPTSTR lpszText) const;

void GetLBText(int nIndex, CString& rString) const;

Записывает содержимое строчки с индексом nIndex в переменные LPTSTR lpszText или CString& rString.

int GetLBTextLen(int nIndex) const;

Возвращает длину строчки с индексом nIndex.

int AddString(LPCTSTR lpszString);

Добавляет строчку в список.

int DeleteString(UINT nIndex);

Удаление строчки с индексом nIndex.

int InsertString(int nIndex, LPCTSTR lpszString);

Заменяет строчку с индексом nIndex содержимым переменной LPCTSTR lpszString.

*Методы класса CProgressCtrl – полоса, дискретно проградуированная слева направо и отображающая процесс выполнения операции*

void SetRange(short nLower, short nUpper);

void SetRange32(int nLower, int nUpper);

Устанавливает минимальное ( nLower ) и максимальное значение ( nUpper ).

void GetRange(int& nLower, int& nUpper);

Записывает в переменные nLower и nUpper минимальное и максимальное значение.

int GetPos();

Возвращает текущее значение.

int SetPos(int nPos);

Устанавливает текущее значение в nPos.

int SetStep(int nStep);

Устанавливает шаг ( nStep ) вывода.

*Методы класса CSliderCtrl – элемент управления, содержащий бегунок, который пользователь может перемещать, выбирая одно из возможных значений*

int GetRangeMax() const;

int GetRangeMin() const;

void GetRange(int& nMin, int& nMax) const;

Первые две функции возвращают максимальное и минимальное знанение, а третья - записывает эти значения в nMax и nMin, соответственно.

void SetRangeMin(int nMin, BOOL bRedraw = FALSE);

void SetRangeMax(int nMax, BOOL bRedraw = FALSE);

void SetRange(int nMin, int nMax, BOOL bRedraw = FALSE);

Первые две функции устанавливают максимальное и минимальное знанение, а третья - устанавливает эти значения из переменных nMax и nMin, соответственно. Аргумент bRedraw отвечает за перерисовку слайдера.

int GetPos() const;

Возвращает текущую позицию.

void SetPos(int nPos);

Устанавливает текущую позицию в nPos.

BOOL SetTic(int nTic);

Устанавливает шаг ( nTic ).

void SetTicFreq(int nFreq);

Устанавливает частоту засечек ( nFreq ).

*Методы класса CSpinButtonCtrl – пара кнопок со стрелками, нажимая которые пользователь может увеличивать или уменьшать некоторое значение*

int SetPos(int nPos);

Устанавливает текущую позицию в nPos.

int GetPos() const;

Возвращает текущую позицию.

void SetRange(int nLower, int nUpper);

void SetRange32(int nLower, int nUpper);

Устанавливает максимальное и минимальное знанение из переменных nMax и nMin, соответственно.

void GetRange(int &lower, int& upper) const;

void GetRange32(int &lower, int& upper) const;

Эти две функции записывают максимальное и минимальное знанение в upper и lower, соответственно.

***Практическая часть***

*Задача.* Разработать простейший телефонный справочник.

*Генерация диалогового приложения с помощью MFC AppWizard*

1. Запустите Microsoft Visual Studio
2. Выполните команду File – New… Выберите вкладку Projects.
3. Выберите тип проекта MFC AppWizard(exe). В поле Location установите рабочую директорию. В поле Project name введите имя проекта (рисунок 121).

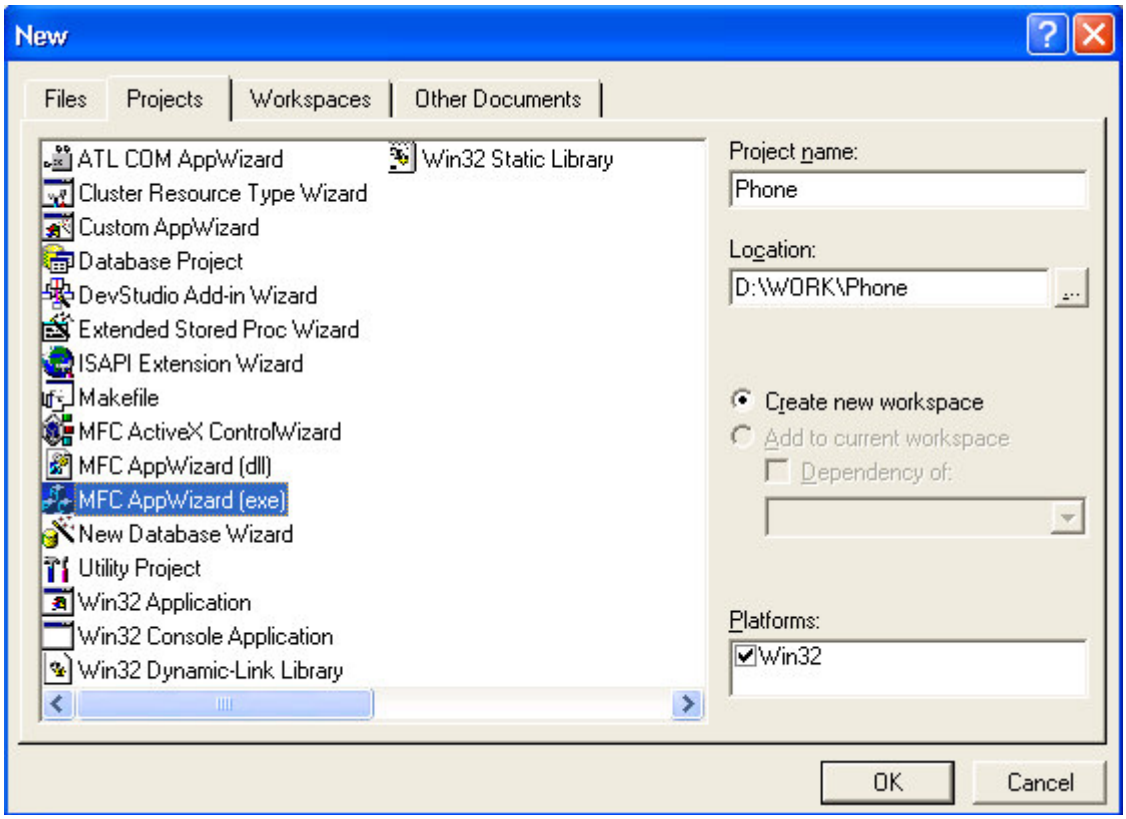



Рисунок 121. Выбор типа проекта

4. Нажмите на кнопку OK. После нажатия кнопки OK инструмент AppWizard будет показывать страницы диалога для уточнения проекта (шаги определения установок проекта). Сделайте выбор в соответствии с таблицей 15.

5. Visual Studio отобразит диалоговое окно New Project Information (информация о новом проекте) – рисунок 122.

Убедитесь, что все установки вашей программы корректны (при необходимости нажмите на кнопу Cancel и внесите изменения). Нажмите на кнопку OK. MFC AppWizard создаст оболочку программы.

Созданные файлы приложения можно посмотреть в окне FileView (рисунок 123).

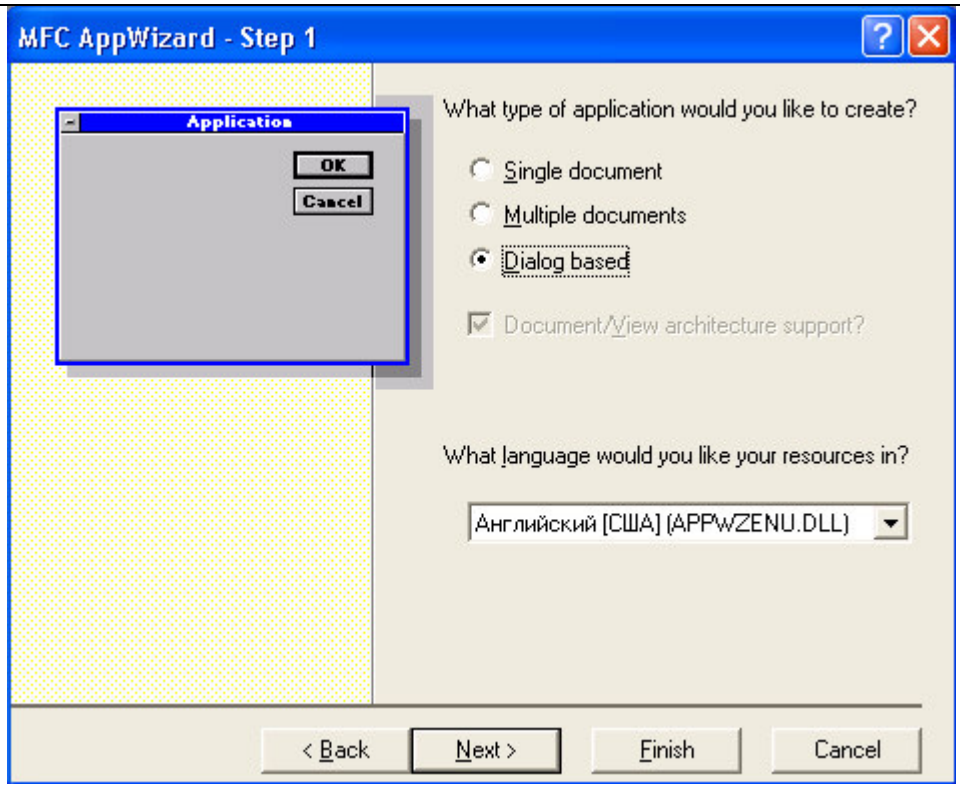
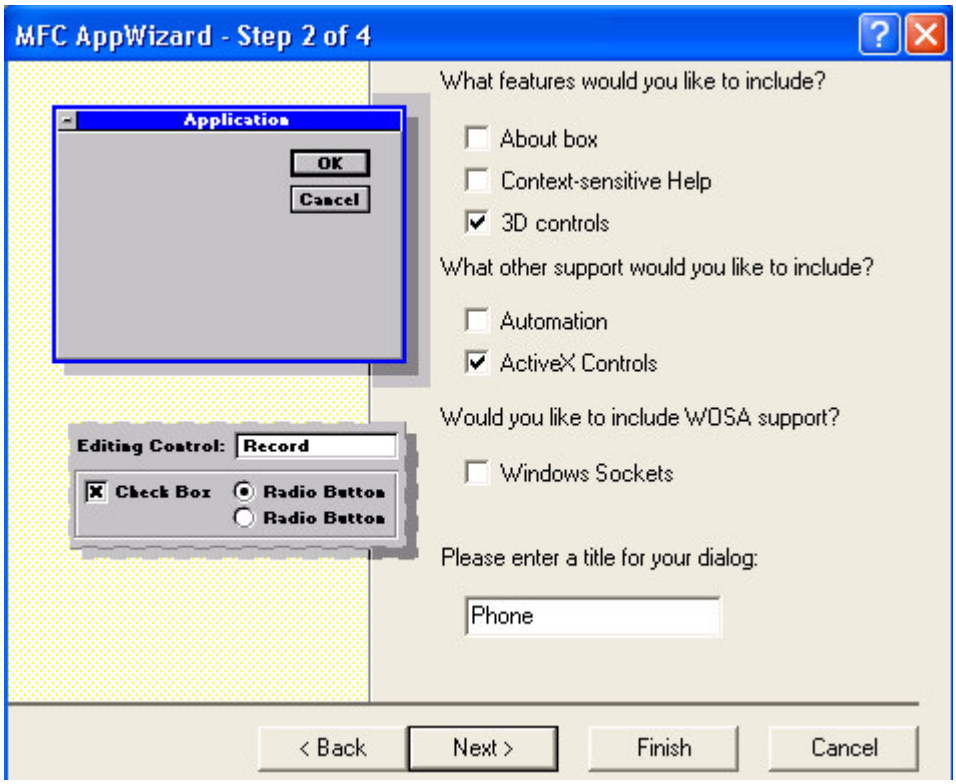
Созданные классы приложения можно посмотреть в окне ClassView. Приложение содержит два класса: класс CPhoneApp (Приложение) и класс CPhoneDlg (производный от CDialog), выполняющий функцию главного окна (рисунок 124).

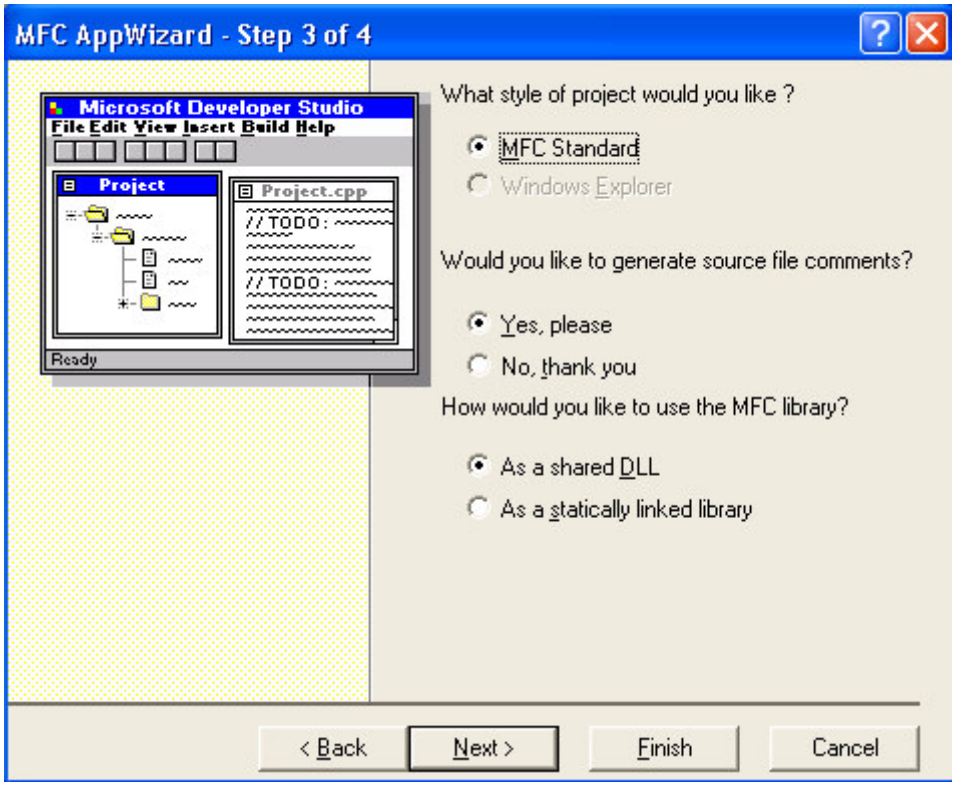
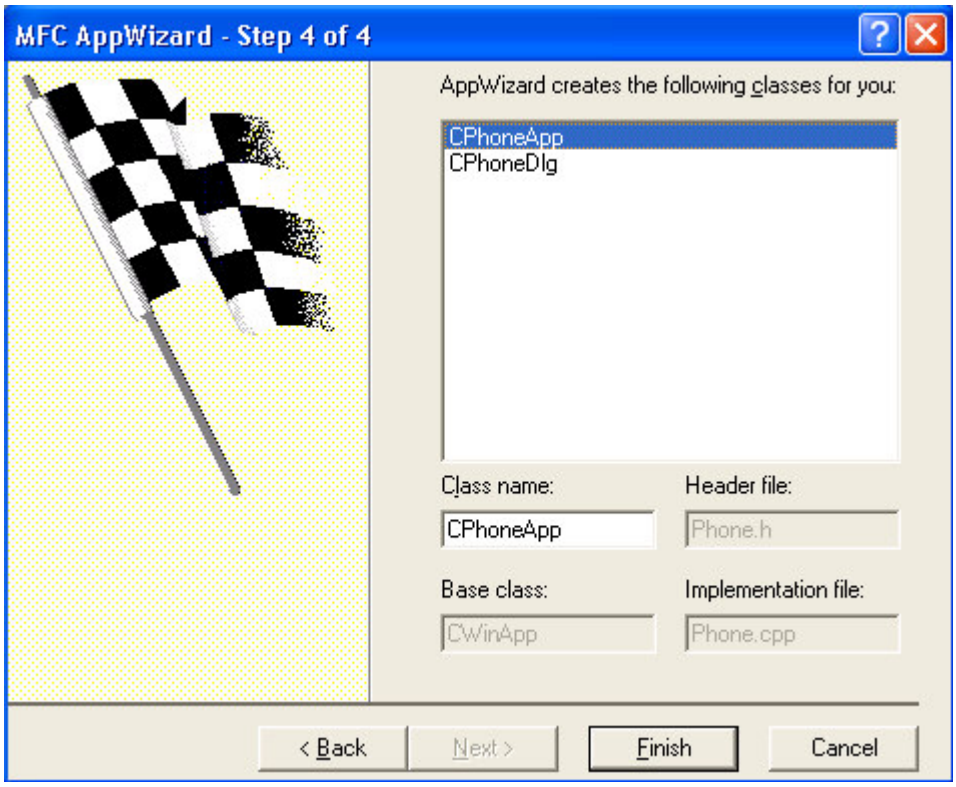
Созданные ресурсы приложения можно посмотреть в окне ResourceView (рисунок 125).

6. Запустите приложение. Приложение готово для диалога с пользователем. Необходимо внести в него желаемую функциональность (рисунок 126).

Таблица 15. Этапы разработки приложения, основанного на диалоге

| Шаг | Выбор или действие |
|---|---|
| 1<br>(Выбор типа приложения) | Выберите переключатель Dialog based (на базе окна диалога). Нажмите на кнопку Next: |
| | |

| Шаг | Выбор или действие |
|---|---|
| |  |
| 2<br>(Настройка возможностей создаваемого приложения) | Выключите About box.<br>В поле Please enter a title for your dialog (Введите заголовок вашего окна диалога) введите Phone. Нажмите на кнопку Next:<br>  |
| 3<br>(Подключение библиотеки MFC) | Выберите переключатели MFC Standard; Yes, please; As a shared Dll. Нажмите на кнопку Next: |

| Шаг | Выбор или действие |
|---|---|
| |  |
| 4<br>(Создание классов приложения) | Нажмите на кнопку Finish:<br>  |

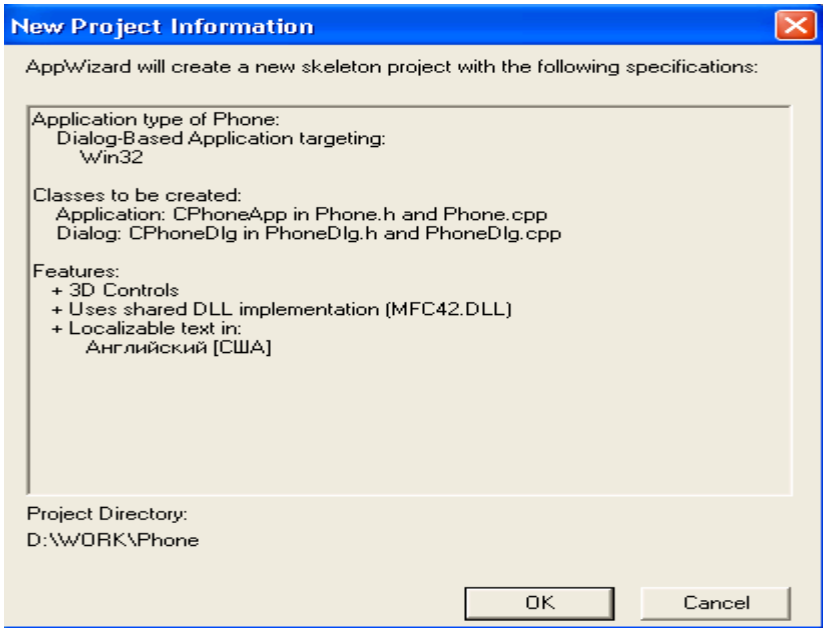



Рисунок 122. Завершение генерации проекта

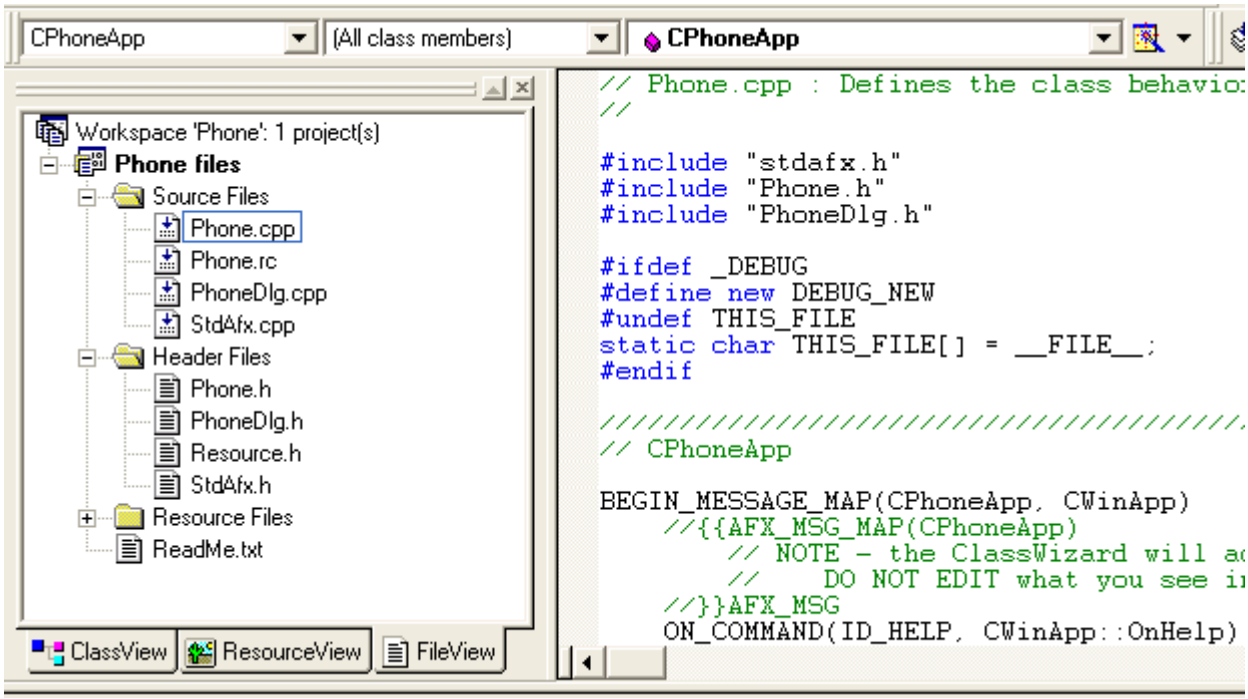



Рисунок 123. Перечень файлов проекта

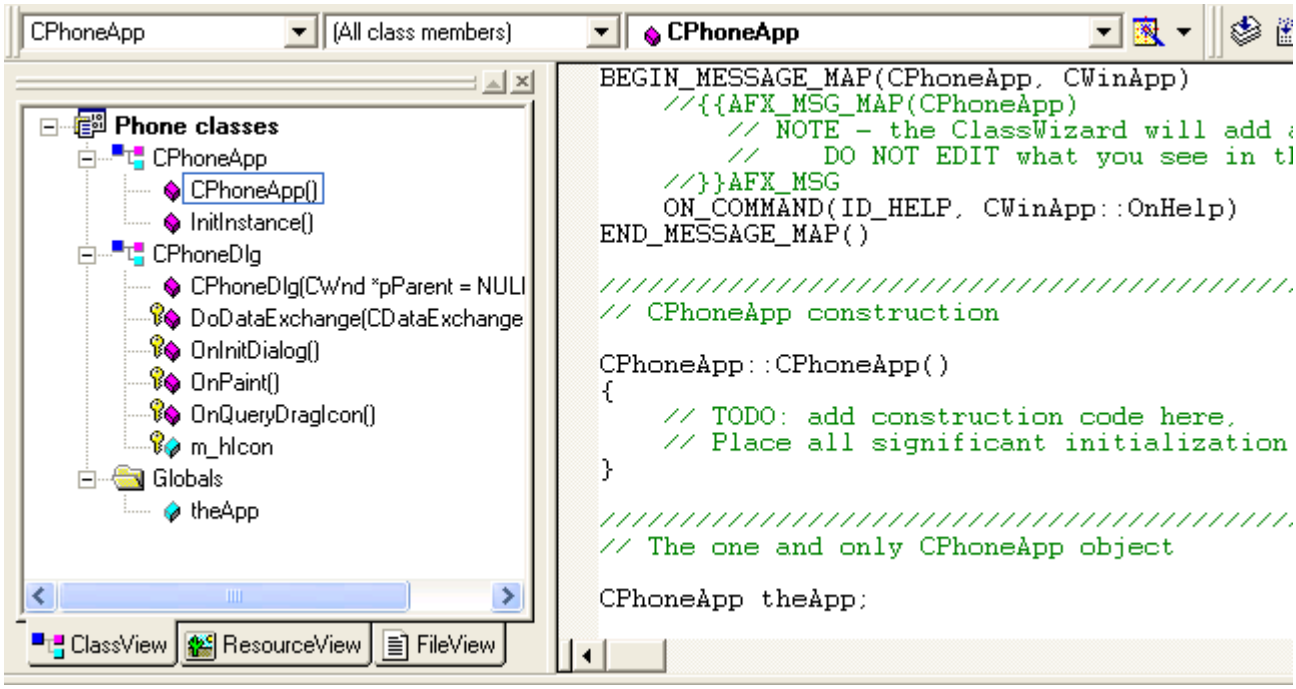



Рисунок 124. Созданные классы приложения

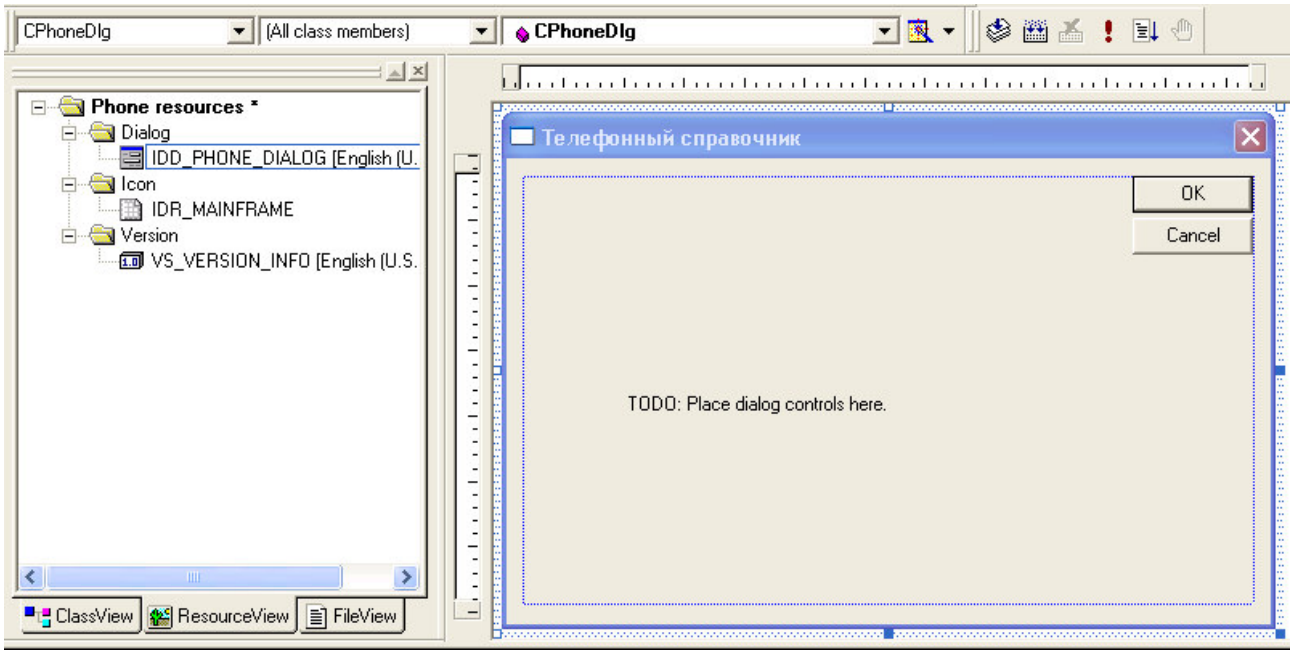



Рисунок 125. Ресурсы приложения

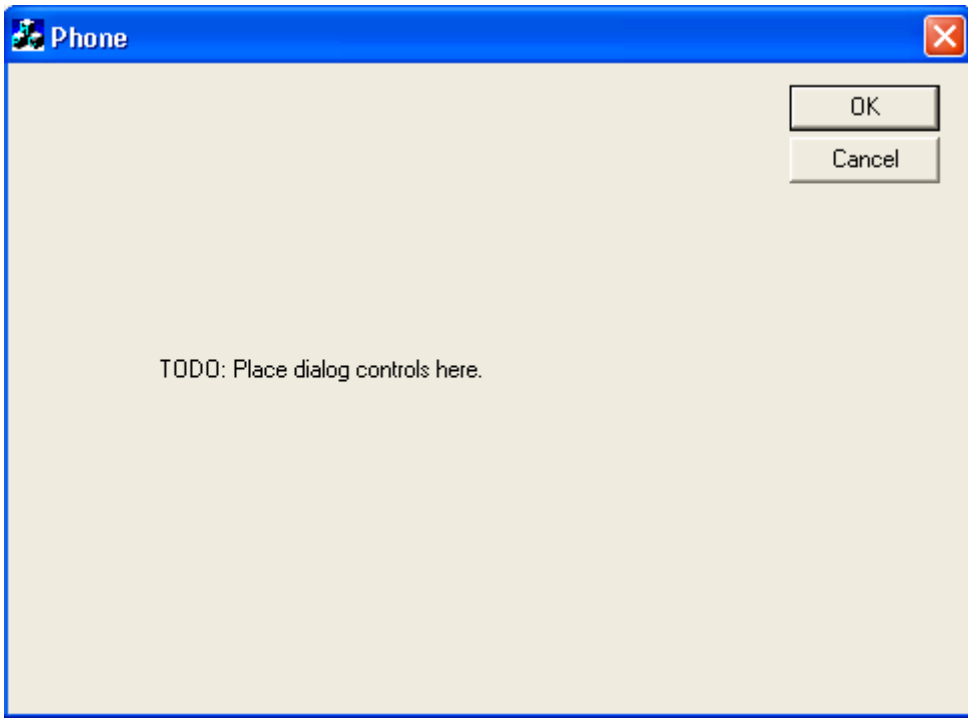



Рисунок 126. Запуск сгенерированного приложения

*Разработка прикладной части приложения*

1. Перейдите в окно ResourceView. Раскройте элемент дерева Dialog. Выделите элемент IDD_PHONE_DIALOG. Для установки русского языка, находясь на этом элементе, в контекстном меню выберите Properties – Language : Russian (рисунок 127).

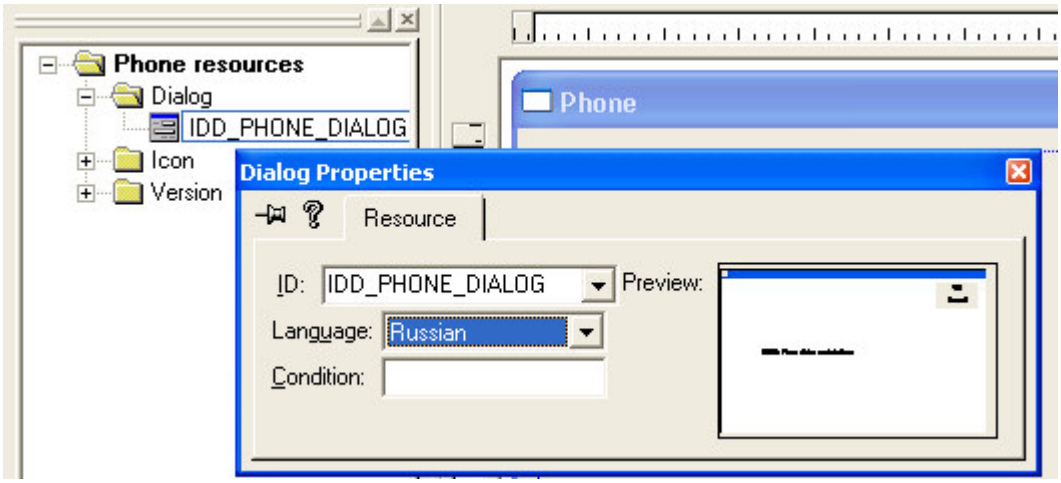



Рисунок 127. Установка русского языка

2. Сохраните изменения с помощью команды File – Save.

3. Измените заголовок окна. Для этого выделите окно. Вызовите контекстное меню, в контекстном меню выберите Properties. Выберите вкладку General. В поле Caption введите заголовок окна «Телефонный справочник» (рисунок 128). Аналогичным образом меняются свойства и других элементов управления.


Рисунок 128. Установка заголовка диалога

4. С помощью панели инструментов Contols вставьте на панель диалога элементы управления в соответствии с рисунком (если панель инструментов Contols не отображается, выполните команду Tools – Customize, в окне Toolbars выберите Contols) (рисунок 129).

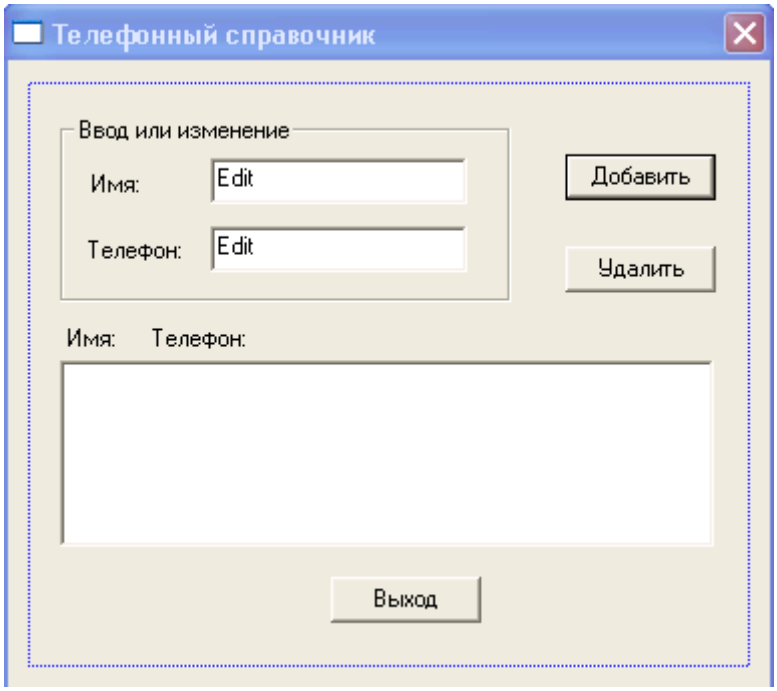



Рисунок 129. Формирование диалогового окна

5. Элементам управления присвойте идентификаторы:

– групповому окну (Group Box): IDC_STATIC;
– полям редактирования (Edit Box): IDC_NAME, IDC_PHONE;
– кнопкам (Button): IDC_ADD (кнопка Добавить), IDC_DEL (кнопка Удалить), IDOK (кнопка Выход);
– окну-списку (List Box): IDC_LIST.

Для кнопки Добавить установите стиль Default button.

6. Введите в класс CPhoneDlg переменные, позволяющие управлять содержимым окон редактирования и окна-списка. Для этого:

– выполните команду View – ClassWizard…;
– выберите опцию Select an existing class;
– укажите имя класса CPhoneDlg, нажмите на кнопку Select;
– выберите вкладку Member Variables;
– выберите идентификатор IDC_NAME и нажмите на кнопку Add Variable;
– заполните поля диалога в соответствии с таблицей:

| Поле | Идентификатор переменной |
|---|---|
| Member Variable name: | sName |
| Category: | Value |
| Variable type: | CString |

– повторите эти действия для поля редактирования IDC_PHONE, задав имя переменной sPhone;
– повторите эти действия для окна-списка IDC_LIST, задав параметры в соответствии с таблицей:

| Поле | Идентификатор переменной |
|---|---|
| Member Variable name: | cList |
| Category: | Control |
| Variable type: | CListBox |

Список переменных будет следующим (рисунок 130).

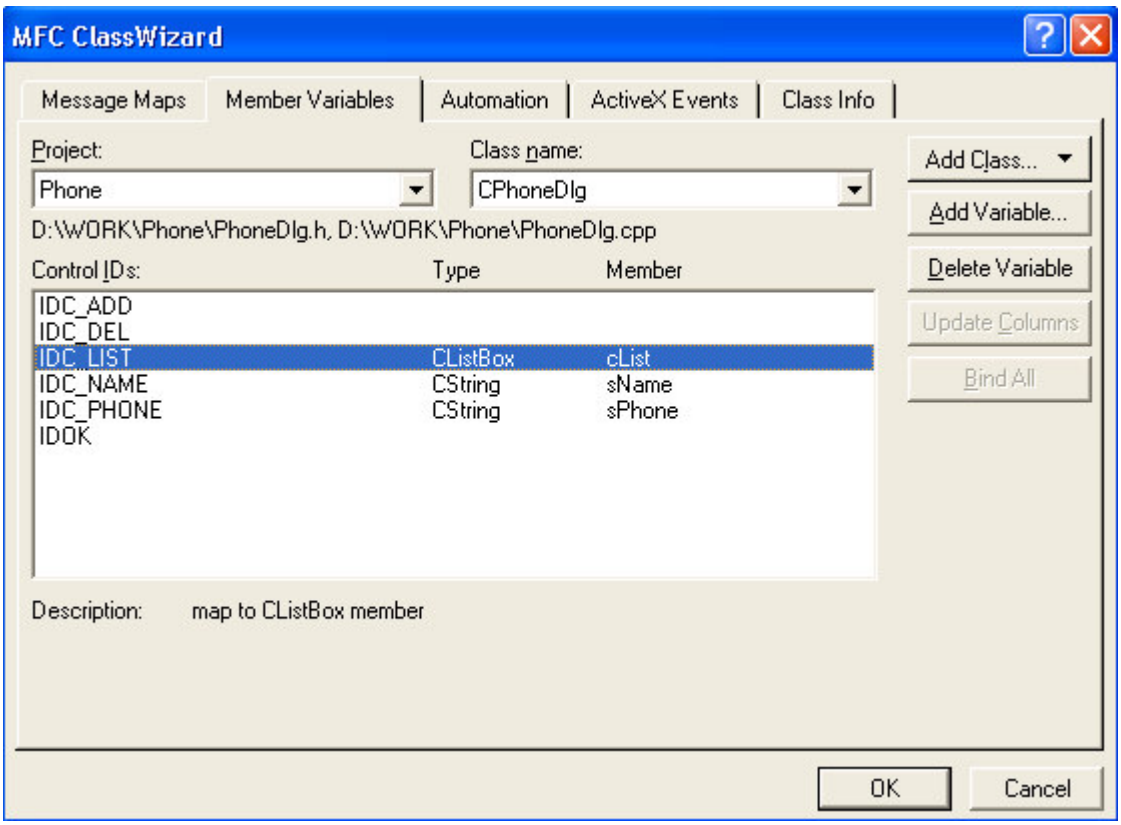



Рисунок 130. Список переменных приложения

7. Задайте функции, которые будут обрабатывать сообщения, возникающие при нажатии кнопок: IDC_ADD (Добавить) и IDC_DEL (Удалить) (рисунок 131). Для этого:
– не выходя из ClassWizard, выберите вкладку Message Maps;
– выберите идентификатор элемента (Object IDs:) IDC_ADD;

– в списке Message: выберите сообщение BN_CLICKED и нажмите на кнопку Add Function;
– согласитесь с предложенным именем (OnAdd) для функции обработчика;
– повторите эти действия для кнопки IDC_DEL;
– согласитесь с введенными изменениями, нажав на кнопку OK:

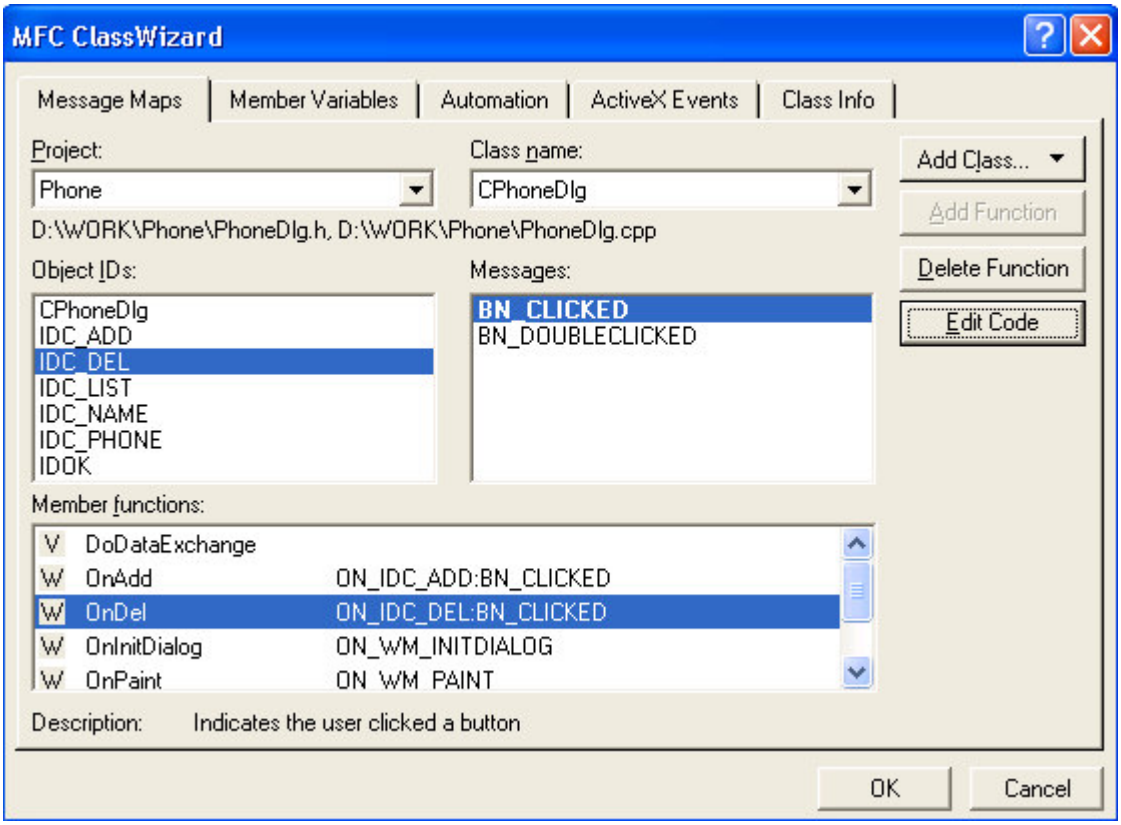



Рисунок 131. Обработчики сообщений приложения

8. Просмотрите коды реализации класса CPhoneDlg, результаты работы ClassWizard.
9. Введите коды метода CPhoneDlg::OnAdd(), сгенерированного ClassWizard:

```
void CPhoneDlg::OnAdd()
{
    UpdateData(TRUE);                    //считывание содержимого полей IDC_NAME и
IDC_PHONE
                                         //формирование строки для списка
    CString add = sName + "  -  " + sPhone;
    cList.AddString(add);       //вставка в список
    sName = "";                          //очистка полей ввода
    sPhone = "";
    UpdateData(FALSE);          //заполнение полей редактирования
}
```

10. Сохраните приложение. Проверьте работу приложения (рисунок 132).

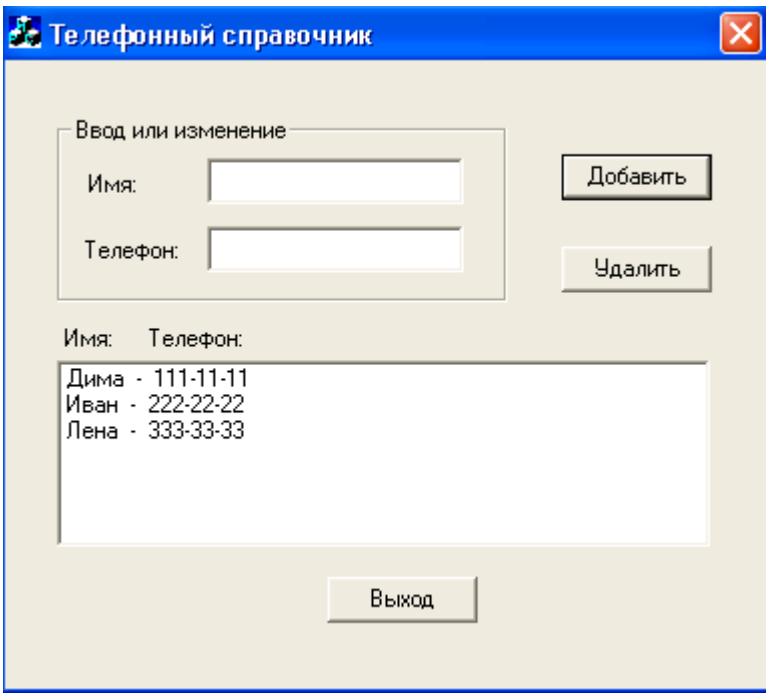



Рисунок 132. Проверка процесса добавления записи

11. Введите коды метода CPhoneDlg::OnDel(), сгенерированного ClassWizard:

```
void CPhoneDlg::OnDel()
{
    int id = cList.GetCurSel();          //индекс выделенного элемента
    cList.DeleteString(id);              //удаление строки
    int last = cList.GetCount()-1;               //индекс последнего элемента
    id = id <= last ? id : id -1;        //индекс элемента, который надо выделить
    cList.SetCurSel(id);                         //выделение элемента
}
```

12. Сохраните приложение. Проверьте работу приложения.

13. Добавим функцию реакции на двойной щелчок мыши на элементе списка. При этом выбранная запись должна быть «расшифрована», а ее составляющие помещены в окна редактирования для возможной коррекции:

- выполните команду View – ClassWizard…;
- выберите вкладку Message Maps;
- выберите идентификатор элемента (Object IDs:) IDC_LIST;
- в списке Message: выберите сообщение LBN_DBLCLK и нажмите на кнопку Add Function;
- согласитесь с предложенным именем (OnDblclkList) для функции обработчика;
- нажмите на кнопку OK

14. Введите коды метода CPhoneDlg::OnDblclkList (), сгенерированного ClassWizard:

```
void CPhoneDlg::OnDblclkList()
{
    CString s;
    int i = cList.GetCurSel();           //индекс выделенного символа
    if (i == LB_ERR)                             //если такого нет, выходим
            return;
    cList.GetText(i,s);                          //считывание текста
    int pos = s.Find('-');                       //поиск разделителя
```

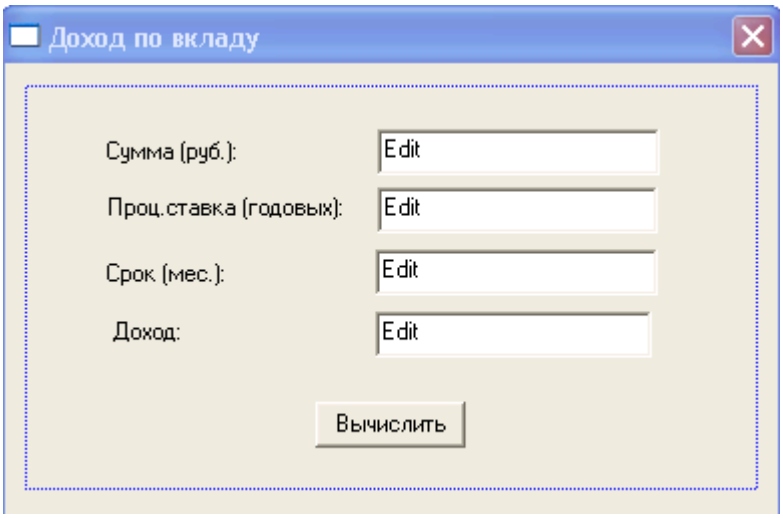
```
    sName = s.Left(pos);                    //выделение имени
    sPhone = s.Mid(pos+1);          //выделение телефона
    sName.TrimRight();                      //удаление лишних пробелов
    sPhone.TrimLeft();
    UpdateData(FALSE);              //заполнение полей редактирования
}
```

15. Сохраните приложение. Проверьте работу приложения.

***Самостоятельная работа***

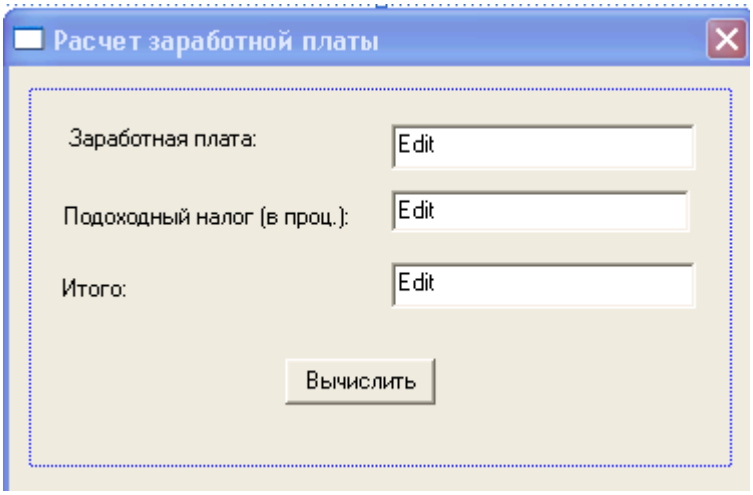
*Вариант 1*

Составить программу на языке C++, которая вычисляет доход по вкладу методом простых процентов (Доход = Сумма * Процент / 12 * Срок). Рекомендуемый вид формы:



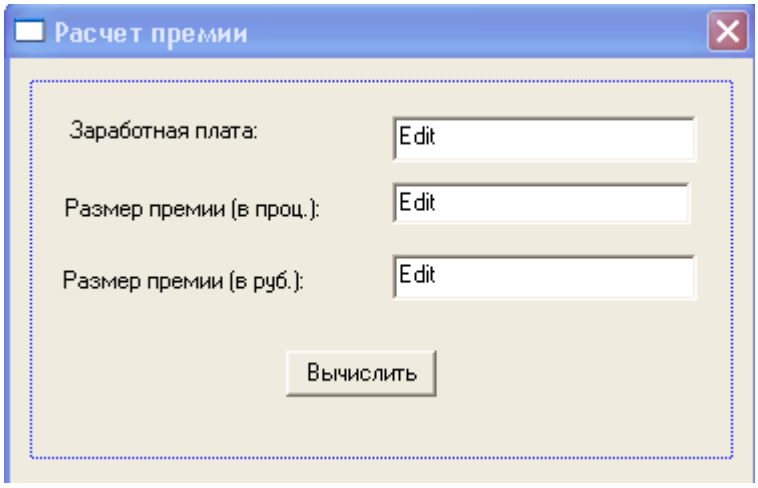

*Вариант 2*

Составить программу «Расчет заработной платы» на языке C++, которая вычисляет заработную плату по формуле: Заработная плата – Налог. Рекомендуемый вид формы:



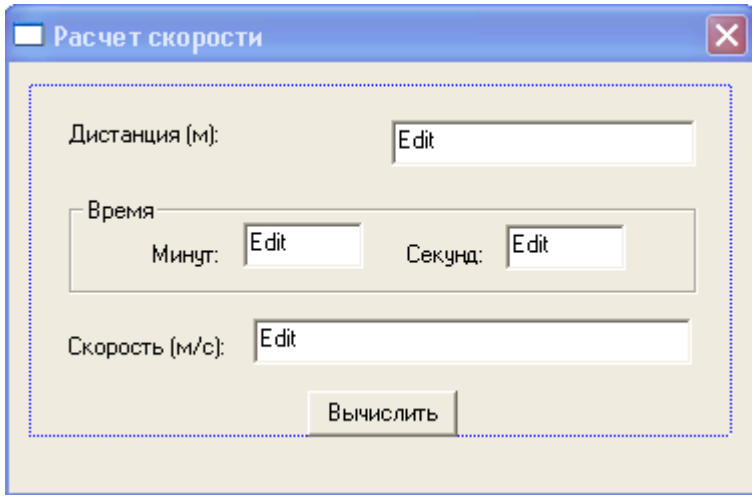

*Вариант 3*

Составить программу на языке C++, которая вычисляет размер премии как заданное число процентов от заработной платы. Рекомендуемый вид формы:



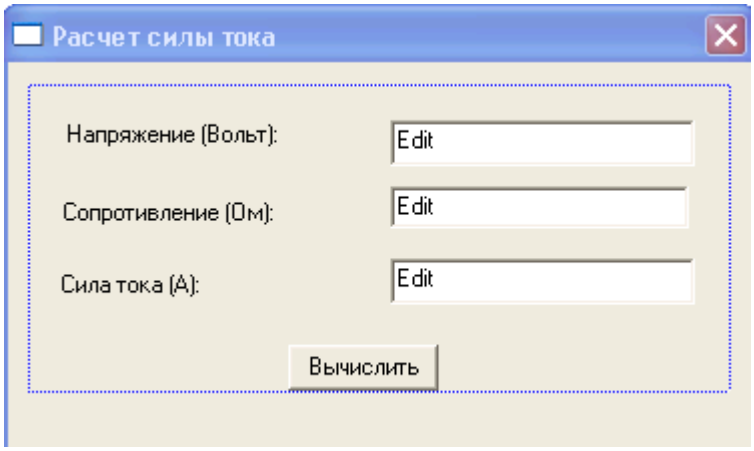

*Вариант 4*

Составить программу на языке C++, вычисляющую скорость (км/ч), с которой бегун пробежал дистанцию. Рекомендуемый вид формы:




*Вариант 5*

Составить программу на языке C++, которая вычисляет силу тока в электрической цепи ( $I = U / R$ ). Рекомендуемый вид формы:



**Лабораторный практикум № 17. Разработка SDI-приложения на языке C++**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 8.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение основных принципов работы с библиотекой классов MFC.

***Вводная часть***

*Архитектура «документ/представление»*

Программы, не использующие архитектуру «документ/представление» имеют некоторые свои преимущества, но такой подход не позволяет в полной мере применять все возможности интерфейса, реализованного библиотекой MFC. В таких программах данные и способ их представления тесно связаны между собой. Так, данные определяются в пределах класса, предназначенного для создания окна. И там же определяется, каким образом данные должны отображаться. Это не всегда желательно, т.к. часто требуется различное представление одних и тех же данных (например, на экране и на принтере). В обычных программах для Windows данные концептуально отделены от способа их представления и часто оба эти понятия смешиваются в одном классе. Но при работе в рамках архитектуры «документ/представление» все организовано иначе. Данные (документ) отделены от своего представления (т.е. вида документа или области просмотра). Это достигается путем инкапсуляции данных в классе, предназначенном для создания документа, и инкапсуляции механизма представления этих данных в классе, предназначенном для организации просмотра документа. Класс просмотра также позволяет управлять вводом данных от пользователя. В программах, построенных на базе архитектуры «документ/представление», объект документа порождается от класса CDocument, а объект представления порождается от класса CView. Представление, создаваемое классом CView, перекрывает главное окно, формируемое классом CFrameWnd.

Основу архитектуры «документ/представление» составляют три глобальных понятия: фрейм (обрамляющее окно), документ и представление.

Документ – это данные, с которыми работает приложение. Термин '"документ" относится к любым видам данных, а не ограничен, например, текстовыми файлами. Представление (вид) отображает данные документа и управляет взаимодействием пользователя с ними. Каждое представление определяет конкретный способ отображения данных. Например, приложение MS Word хранит документ в виде doc-файла, который при просмотре из Far или Windows Commander имеет много вспомогательной информации, а представление этих данных на экране имеет вполне определенный читабельный вид. С другой стороны, это же приложение совсем иначе отображает этот же документ при предварительном просмотре перед печатью.

С каждым документом может быть связано несколько представлений, но каждое представление связано только с одним документом. Если приложение может одновременно использовать только один документ, то говорят, что оно имеет однодокументный интерфейс (SDI – Single Document Interface). Если оно одновременно может открыть несколько документов, то говорят, что приложение имеет многодокументный интерфейс (SDI – Multiple Document Interface). Классическим примером SDI-приложения является редактор Notepad, а MDI-приложения – MS Word.

Архитектура «документ/представление» охватывает следующие основные классы (кроме классов для работы с базами данных и OLE):

• CWinApp – класс для создания приложения;

• CFrameWnd – класс для создания главного окна однодокументного приложения и базовый для классов CMDIFrameWnd и CMDIChildWnd, которые отвечают за работу многодокументного приложения;

• CDocTemplate – базовый абстрактный класс для создания шаблонов документов; при работе с однодокументным приложением используется производный от него класс CSingleDocTemplate, а для многодокументных - класс CMultiDocTemplate;

• CDocument – класс для создания документа;

• CView – базовый класс, который совместно со своими производными классами – CCtrlView, CEditView, CListView, CTreeView и CScrollView – отвечают за отображение данных документа и за взаимодействие с пользователем; для этих же целей можно использовать класс CSplitterWnd.

На рисунке 133 показано соотношение документа и представления.

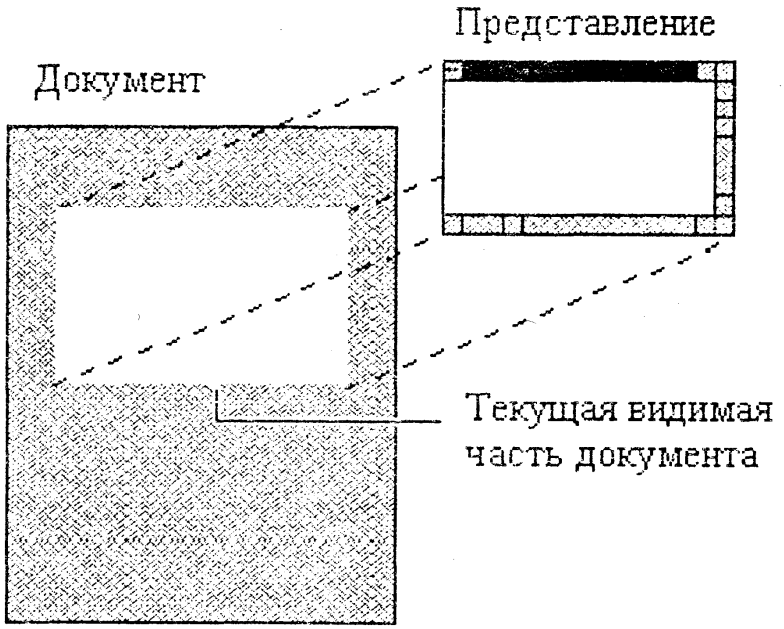



Рисунок 133. Соотношение документа и представления

*Место объекта-приложения в архитектуре «документ/представление»*

Приложение может поддерживать произвольное число типов документов. Для этого надо создать и зарегистрировать во время инициализации объекта-приложения (при выполнении функции InitInstance()) нужное число документов. Для каждого типа документа, с которым предполагается работа, используется свой шаблон.

В программах, работающих на базе архитектуры «документ/представление», все объекты являются динамическими, т.е. объекты главного окна, документа и представления создаются динамически, так как они должны периодически загружаться с диска или создаваться заново. Для этого используются две макрокоманды: DECLARE_DYNCREATE() и IMPLEMENT_DYNCREATE().

В библиотеке MFC для работы с шаблонами документов реализованы класс CDocTemplate и два производных от него класса – CSingleDocTemplate и CMultiDocTemplate.

Иерархия классов для классов работы с шаблоном документов представлена на рисунке 134.

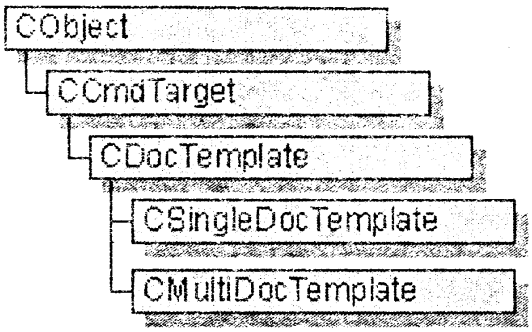


Рисунок 134. Иерархия классов для классов работы с шаблоном документов

Класс CDocTemplate абстрактный. В нём реализованы основные функциональные возможности для работы с шаблонами документов – организация и управление взаимодействием между классами трех типов:

• классом документа, образованного из CDocument;

• классом представления, который изображает данные документа. Можно создать этот класс на базе CView, CScrollView, CFormView, CEditView и т.д.;

• классом фрейма, который содержит представление. Для SDI-приложений он создается на основе CFrameWnd, а для MDI-приложений – на базе CMDIChildWnd.

*Класс CSingleDocTemplate*

Класс CSingleDocTemplate определяет шаблон документа для однодокументного интерфейса. В SDI-приложениях главный фрейм является одновременно и фреймом документа, т.е. в определённый момент времени может быть открыт только один документ.

Конструктор класса имеет следующий вид:

```
CSingleDocTemplate(
    UINT nIDResource,
    CRuntimeClass *pDocClass,
    CRuntimeClass *pFrameClass,
    CRuntimeClass *pViewClass);
```

nIDResource – идентификатор ресурсов, используемых с этим типом документов.

pDocClass – указатель на объект CRuntimeClass, отвечающий за данные документа.

pFrameClass – указатель на объект CRuntimeClass, характеризующий фрейм документа.

pViewClass – указатель на объект CRuntimeClass, отвечающий за представление документа.

Кроме конструктора, в классе реализованы четыре «чистые» функции класса CDocTemplate.

```
virtual POSITION
CDocTemplate::GetFirstDocPosition();
```

– возвращает позицию первого документа из списка документов, ассоциированных с данным шаблоном, или NULL, если список пуст. Функция переопределена в обоих производных классах -

CSingleDocTemplate и CMultiDocTemplate.

```
virtual CDocument*
CDocTemplate::GetNextDoc(POSITION& rPos);
```

– возвращает указатель на объект-документ из общего списка документов, ассоциированных с шаблоном, который хранится непосредственно за документом, заданным его позицией в списке (параметр rPos).

```
virtual CDocument*
CDocTemplate::OpenDocumentFile(LPCTSTR lpszPathName,
BOOL bMakeVisible = TRUE);
```

– позволяет открыть файл, заданный параметром lpszPathName. Если его значение равно NULL, то вызывается функция CreateNewDocument(), которая создает новый файл.

```
virtual void
CDocTemplate::SetDefaultTitle(CDocument* pDocument);
```

– загружает заголовок документа, используемый по умолчанию и определенный в строке ресурса.

Кроме перечисленных «чистых» функций, в классе переопределены также функции базового класса AddDocument() и RemoveDocument().

*Класс CMultiDocTemplate*

Этот класс определяет шаблон документа для многодокументного интерфейса. MFC-приложения используют главный фрейм как рабочее место, в котором пользователь может открывать произвольное число окон документа, отображающих данные того или иного документа.

Приложение использует шаблоны документов, когда пользователь создает новый документ. Если приложение поддерживает больше одного типа документов, то библиотека получает имена поддерживаемых

типов из шаблона и отображает их в списке блока диалога New, реализованного на базе класса CNewTypeDlg. Как только пользователь выбрал тип документа, приложение создает объекты «документ», «фрейм» и «представление» и сопоставляет их друг другу.

В классе определены два общедоступных члена, отвечающих за совместное использование меню и таблицы акселераторов:

HMENU m_hMenuShared;

HACCEL m_hAccelTable;

Конструктор класса:

```
CMultiDocTemplate(
    UINT nIDResource,
    CRuntimeClass *pDocClass,
    CRuntimeClass *pFrameClass,
    CRuntimeClass *pViewClass);
```

Кроме конструктора, в классе реализованы четыре «чистые» функции базового класса:

GetFirstDocPosition(), GetNextDoc(),

OpenDocumentFile(), SetDefaultTitle().

и добавлены три функции:

LoadTemplate(), AddDocument(), RemoveDocument().

*Документы и представления*

Документ представляет собой некоторую единицу данных, которую пользователь обычно открывает по команде ID_FILE_OPEN и сохраняет по команде ID_FILE_SAVE.

Для реализации документа в типичном приложении надо проделать следующую последовательность действий:

• для каждого типа документа образовать класс на базе CDocument;

• добавить в класс CDocument переменные для хранения всех данных документа;

• реализовать функцию для чтения и модификации этих данных;

• переопределить функцию CObject::Serialize() в новом классе документа для организации чтения/записи данных документа с диска/на диск.

Последовательность действий при создании документа приведена на рисунке 135.

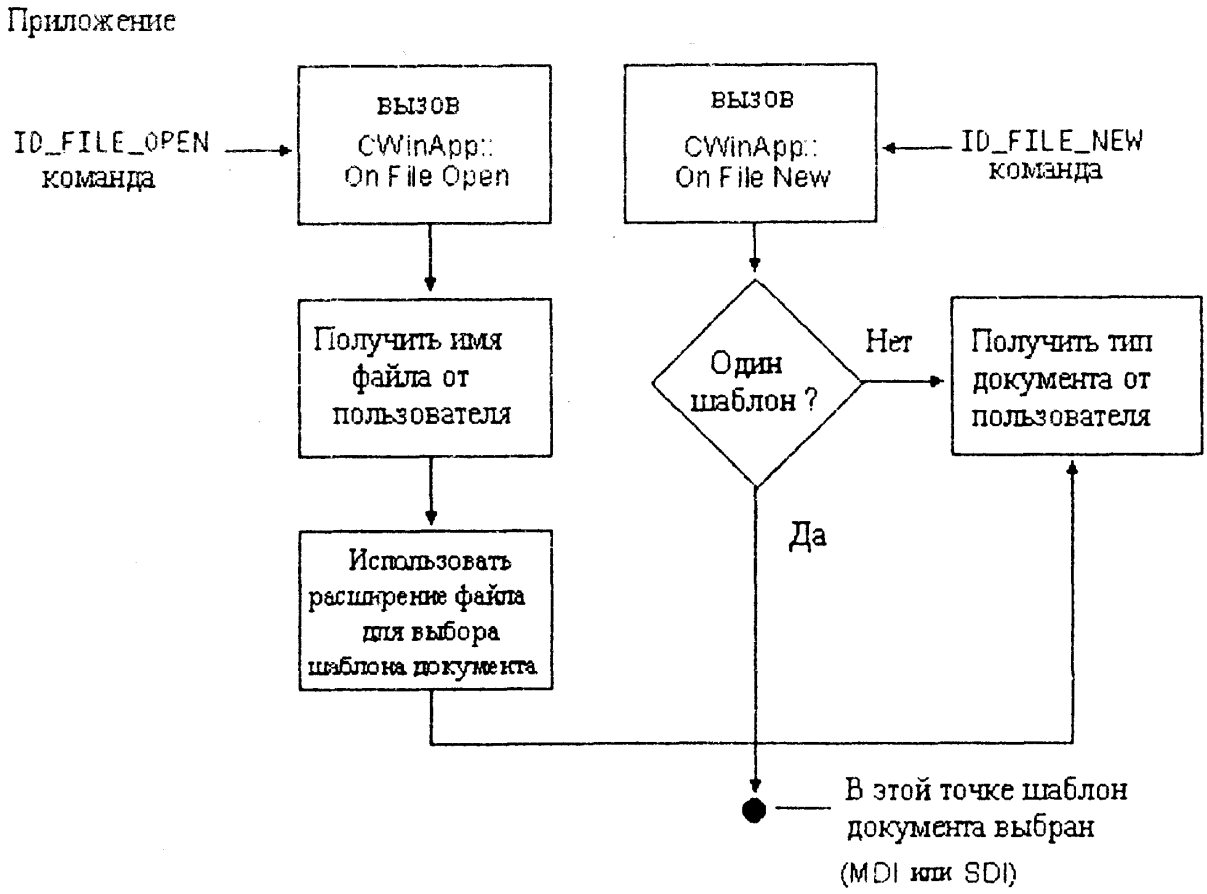



Рисунок 135. Последовательность действий при создании документа

Данные документа определяются как переменные специального класса документа, производного от CDocument. Помимо данных, в классе документа часто определяются специальные функции для установки и извлечения элементов данных и для выполнения необходимых операций над ними. Функция Serialize() отвечает за процесс сериализации.

Опишем жизненный цикл документа в рамках архитектуры «документ/представление»:

• во время динамического создания вызывается конструктор объекта – «документ»;

• для каждого нового документа вызывается функция OnNewDocument() или OnOpenDocument();

• пользователь взаимодействует с документом посредством представлений, ассоциированных с ним;

• для удаления данных документа вызывается функция DeleteContents();

• для удаления объекта «документ» вызывается его деструктор.

Функции класса CDocument представлены в таблице 16.

Таблица 16. Функции класса CDocument

| Функция | Описание |
|---|---|
| GetPathName() | Возвращает полный путь к файлу документа |
| SetPathName() | Устанавливает полный путь к файлу документа |
| GetTitle() | Возвращает название документа |
| SetTitle() | Устанавливает название документа |
| AddView() | Присоединяет представление к документу |
| RemoveView() | Отсоединяет представление от документа |
| UpdateAllView() | Информирует все представления, присоединенные к документу о том, что документ был изменен. (Вызывает для каждого представления функцию OnUpdate()) |

| | |
|---|---|
| OnOpenDocument() | Вызывается библиотекой MFC при обработке команды ON_FILE_OPEN |
| OnSaveDocument() | Вызывается библиотекой MFC при обработке команды ON_FILE_SAVE |
| GetFile() | Возвращает указатель на объект CFile по заданному пути к файлу |
| ReleaseFile() | Освобождает файл |
| OnCloseDocument() | Вызывается, когда пользователь закрывает документ. В классе CDocument есть флаг изменения. Его значение зависит от того, был ли изменен документ с момента его последнего сохранения. За установку флага отвечает программа. Для этого используется метод<br>SetModifiedFlag(), принимающий на входе логическую переменную. Опрос флага выполняется функцией IsModified() |
| PreCloseFrame() | Вызывается библиотекой MEC до разрушения фрейма |

Представления – специальная группа классов дочерних окон фрейма, которая отвечает за отображение данных документа и за взаимодействие с пользователем.

Представления ассоциируется с документом и действует как посредник между ним и пользователем, т.е. представление переводит образ документа на экран, принтер или любое другое устройство графического вывода, и интерпретирует действия пользователя как операции над документом. Обратите внимание, что представление может быть ассоциировано только с одним документом. С другой стороны, в рамках одного фрейма документа можно создавать и использовать сколько угодно представлений для работы с одним и тем же документом.

Действия по созданию фрейма и представления приведены на рисунках 136 и 137.

Опишем кратко классы представлений.

CView – класс предоставляет базовые функциональные возможности для всех классов представлений, которые есть в библиотеке или определяются пользователем.

CCtrlView – базовый класс для классов элементов управления, включающих CEditView, CListView, CRichEditView и CTreeView, которые адаптированы к архитектуре «документ/представление».

CEditView – определяет представление, которое, подобно классу CEdit, инкапсулирует функциональные возможности элемента управления EDIT и может быть использовано для реализации простейшего текстового редактора.

CScrollView – берет на себя всю работу по обеспечению автоматической прокрутки и масштабирования.

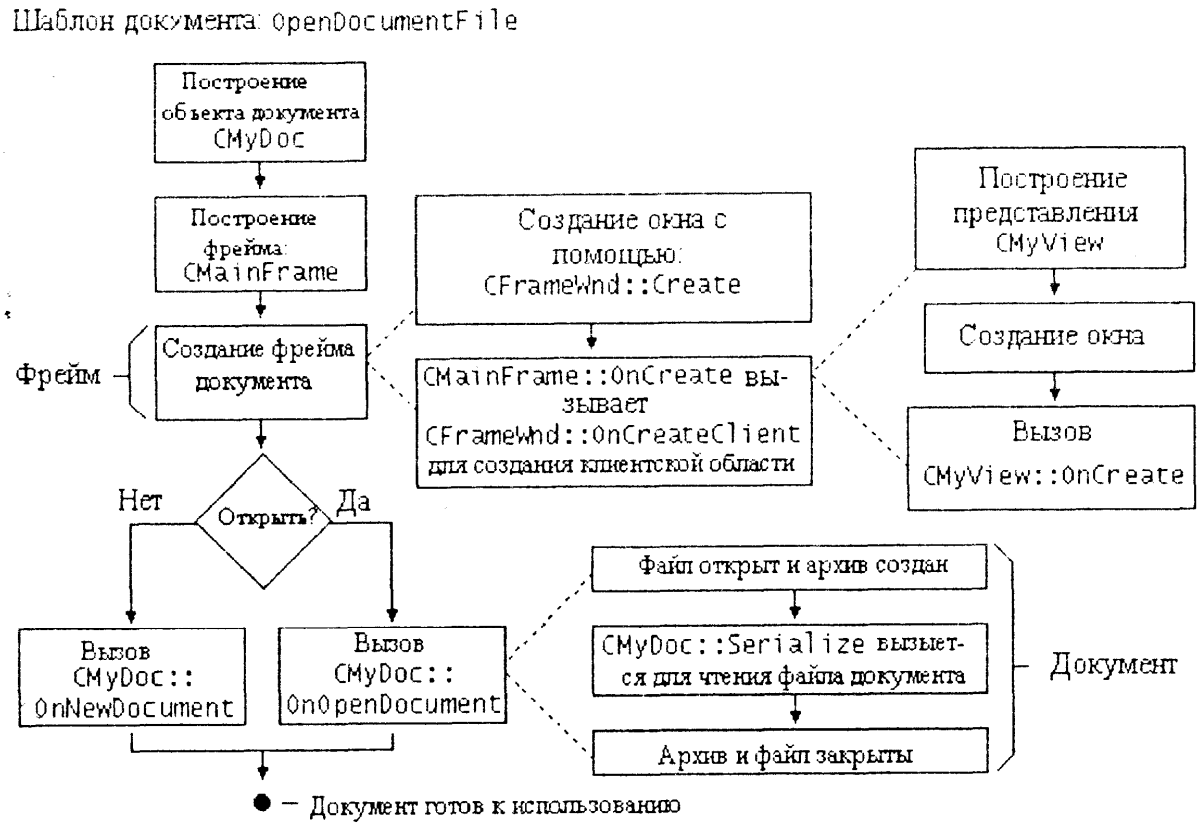



Рисунок 136. Действия по созданию фрейма

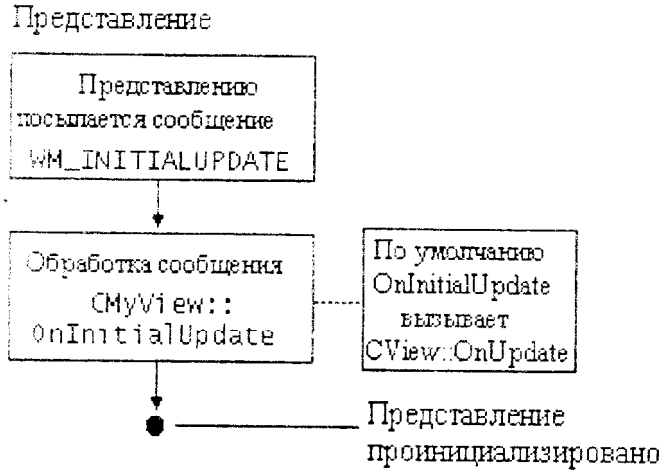



Рисунок 137. Действия по созданию представления

Класс CView содержит большое число функций. Одна из наиболее часто используемых функций – GetDocument():

CDocument* GetDocument() const;

Она возвращает указатель на объект класса CDocument, связанный с представлением.

Функция используется, когда представление должно получить доступ к данным, хранящимся в документе.

Virtual void OnDraw(CDC *pdc) = 0;

Эта чисто виртуальная функция обязательно должна быть переопределена для изображения представления документа. Это сделано в производных классах. Данная функция используется вместо обработчика сообщения WM_PAINT и получает как параметр указатель на готовый контекст устройства. Библиотека MFC использует эту функцию как для печати и предварительного просмотра документа, так и для отображения его на экране.

Для получения родительского окна используется функция:

CFrameWnd* CWnd::GetparentFrame() const;

Функция возвращает указатель на обрамляющее окно, иначе NULL.

*Сохранение и загрузка документов*

Документы можно сохранять на диске и загружать их оттуда. Этому способствует два обстоятельства. Во-первых, класс CObject содержит функцию Serialize(), которая автоматически вызывается при сохранении или загрузке документа, Во-вторых, в MFC есть класс CArchive, предназначенный для автоматизации почти всех функций работы с файлами. В классе CArchive есть перегруженные операторы ввода-вывода - << и >>, предназначенные для работы и со стандартными типами данных и с типами классов, определённых в иерархии MFC. Механизм, обеспечивающий хранение документов в архитектуре «документ/представление», называется идентификацией.

Функция CObject::Serialize() вызывается при загрузке или сохранении документов:

virtual void Serialize (CArchive& ar);

Параметр функции определяет поток архивирования данных. Для различения – сохраняется документ или загружается – служат функции:

BOOL CArchive::IsLoading() const;

BOOL CArchive::IsStoring() const;

Первая возвращает TRUE, если документ загружается, иначе FALSE. А функция IsStoring() – наоборот, возвращает TRUE, если сохраняется, иначе FALSE.

Для записи или чтения документа в функции Serialize используются операторы << и >>. В качестве левого операнда этих операторов применяется параметр ar.

***Практическая часть***

*Задача.* Разработать простейший телефонный справочник.

*Генерация SDI-приложения с помощью MFC AppWizard*

1. Запустите Microsoft Visual Studio.
2. Выполните команду File – New… Выберите вкладку Projects.
3. Выберите тип проекта MFC AppWizard(exe). В поле Location установите рабочую директорию. В поле Project name введите имя проекта (рисунок 138).
4. Нажмите на кнопку OK. После нажатия кнопки OK инструмент AppWizard будет показывать страницы диалога для уточнения проекта (шаги определения установок проекта). Сделайте выбор в соответствии с таблицей 17.
5. Visual Studio отобразит диалоговое окно New Project Information (информация о новом проекте) (рисунок 139).

Убедитесь, что все установки вашей программы корректны (при необходимости нажмите на кнопу Cancel и внесите изменения). Нажмите на кнопку OK. MFC AppWizard создаст оболочку программы.

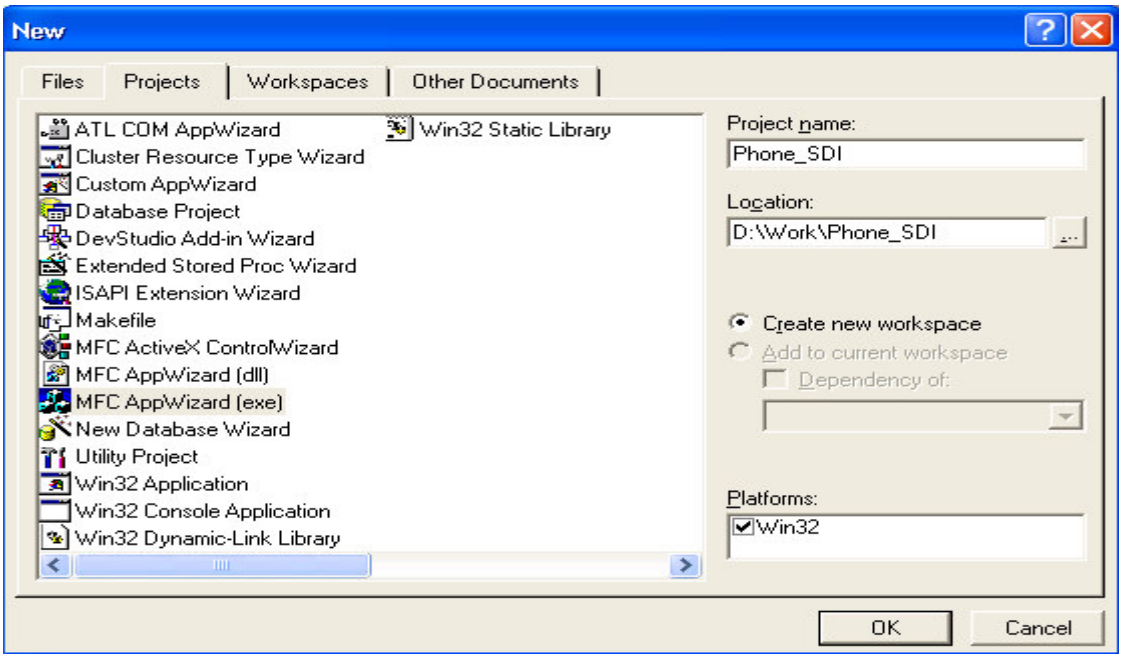
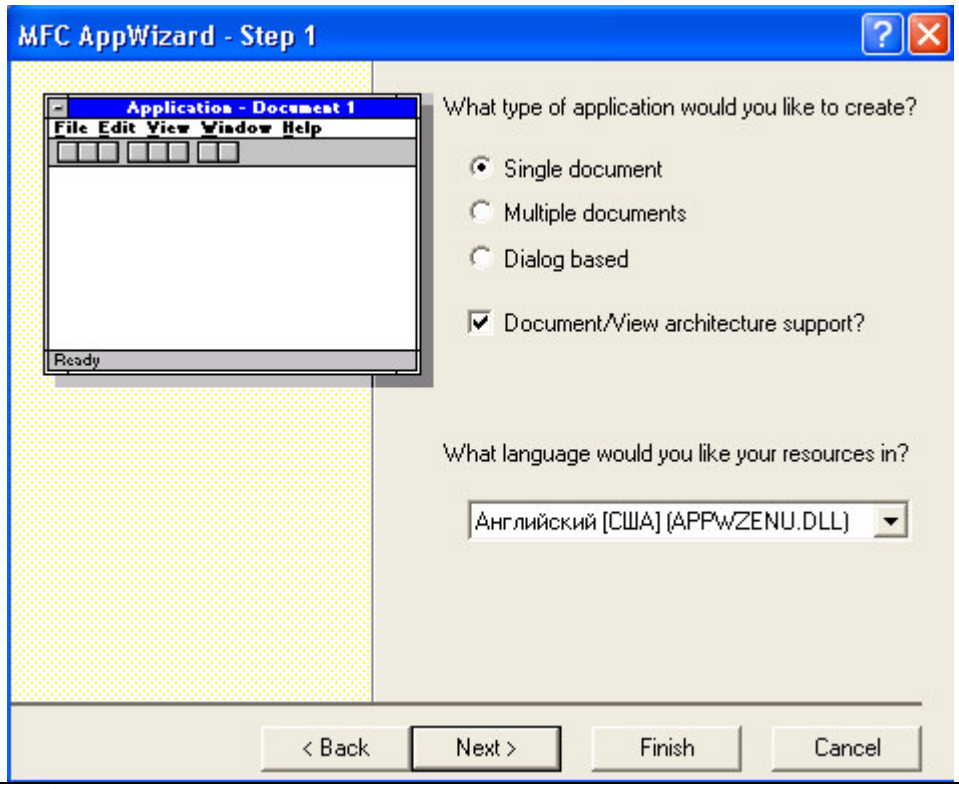
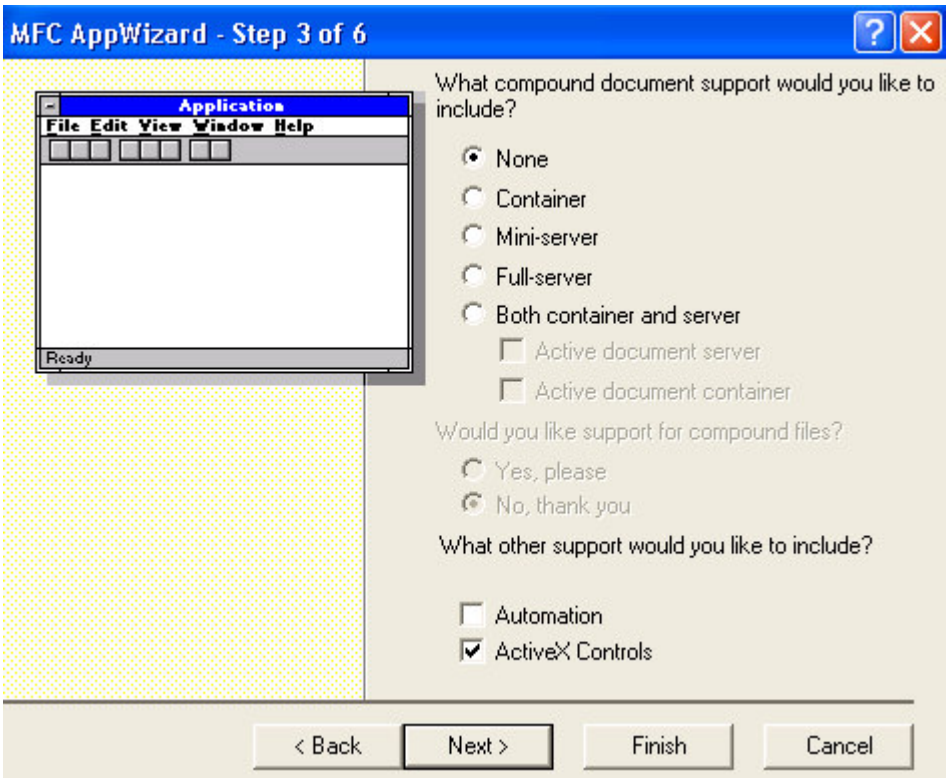
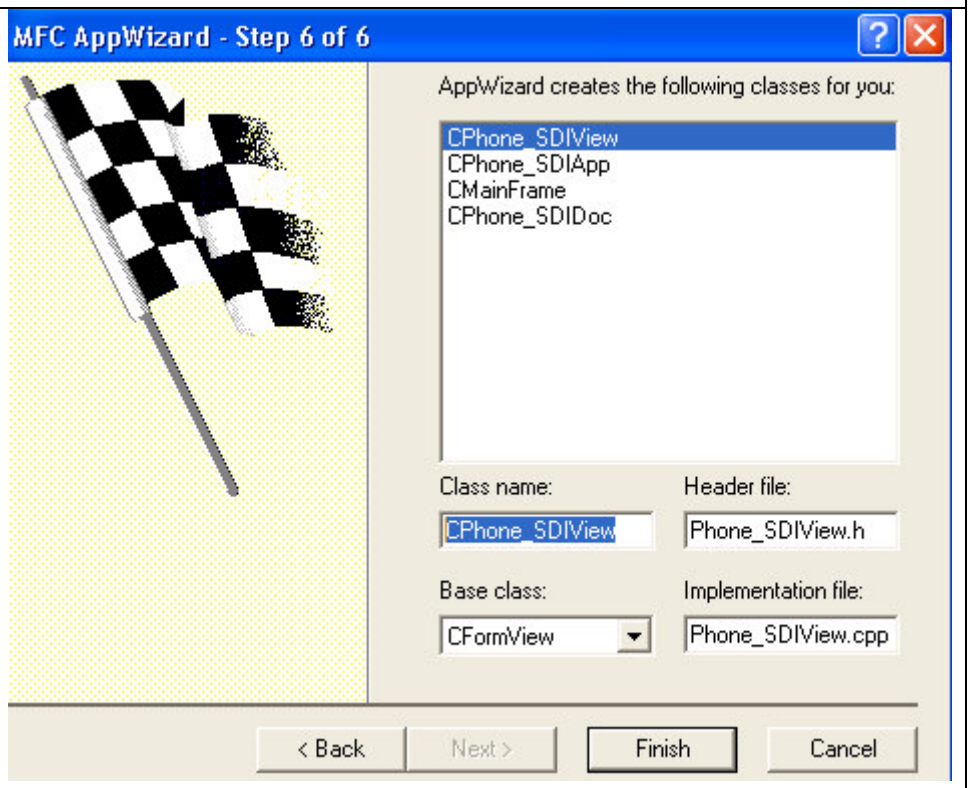



Рисунок 138. Выбор типа приложения

Таблица 17. Этапы разработки SDI-приложения

| Шаг | Выбор или действие |
|---|---|
| 1<br>(Выбор типа приложения) | Выберите переключатель Single document (однодокументный интерфейс). Нажмите на кнопку Next:<br>  |
| 2<br>(Поддержка базы данных) | Выберите None. Нажмите на кнопку Next: |

| Шаг | Выбор или действие |
|---|---|
| 3 (Поддержка составных документов) | Выберите None. Нажмите на кнопку Next:<br>  |
| 4 (Включение свойств) | Нажмите на кнопку Advanced: |
| 5 (Подключение библиотеки MFC) | Выберите переключатели MFC Standard; Yes, please; As a shared Dll. Нажмите на кнопку Next: |
| 6 (Создание классов приложения) | Выберите для класса CPhone_SDIView базовый класса CFormView. Нажмите на кнопку Finish: |

| Шаг | Выбор или действие |
| --- | --- |
| |  |



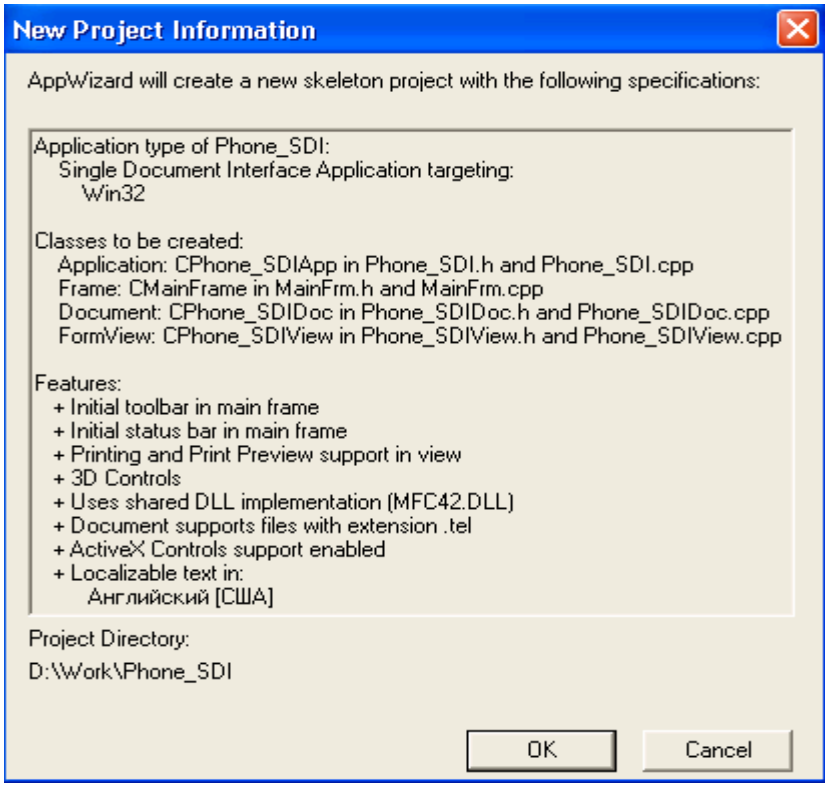




Рисунок 139. Завершение генерации приложения

Созданное приложение состоит из пяти классов:

• CPhone_SDIApp – класс приложения, поддерживающий функциональность SDI-приложения;

• CPhone_SDIDoc – класс Документ, поддерживающий функциональность документа;

• CPhone_SDIView – класс Вид, поддерживающий один из многих возможных представлений документа;

• CMainFrame – класс главного окна приложения или окна-рамки (Frame window);

• CAboutDlg – класс диалога About меню Help.

Все эти классы являются производными от классов MFC. У всех них есть общие предки – классы CObject и CCmdTarget (последний является потомком первого). Наследование данных и методов, поддерживаемое языком C++, означает, что все пять классов приложения унаследовали функциональность этих классов. Например, CCmdTarget передал классам приложения способность реагировать на команды пользователя и сообщения Windows, обрабатывая их в функциях-обработчиках (Message handlers).

Класс CObject обеспечивает каждому из классов следующие возможности:

• совместимость с классами семейства Collection. Классы этого семейства умеют эффективно управлять фундаментальными динамическими структурами данных (массивы, списки и отображения объектов произвольного типа);

• получение информации об используемых типах на этапе выполнения (Runtime class information) и динамическое создание объектов (Dynamic creation);

• диагностический вывод;

• устойчивость объектов – способность самостоятельно сохранять свои данные в долговременной памяти.

6. Запустите приложение. SDI-проект отличается тем, что он поддерживает однодокументный интерфейс (рисунок 140). Запущенное приложение имеет меню, панель инструментов и строку состояния. Опробуйте все команды меню. Приложение готово выполнять файловые операции и пытается найти файлы с расширением tel. Выберите команду File - Save и запишите документ под каким-нибудь именем, нажмите кнопку New и вновь запишите под другим именем. Вы увидите, что пустые файлы типа tel сохраняются, а их список отслеживается в меню File – Recent File.

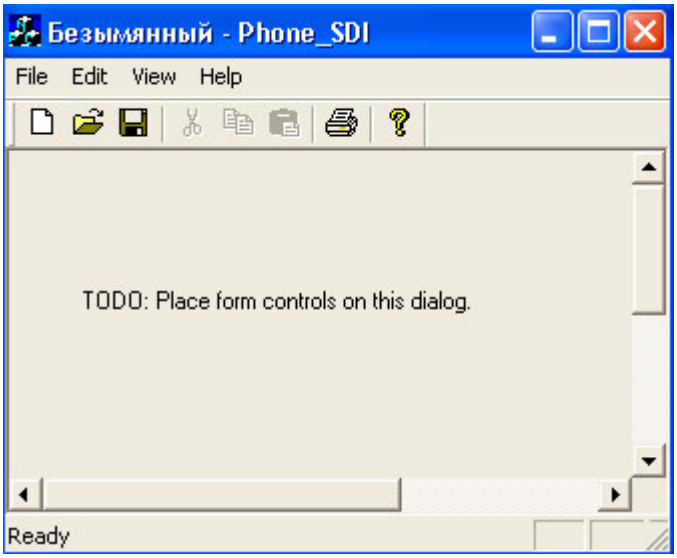



Рисунок 140. Запуск сгенерированного приложения

***Разработка прикладной части приложения***

1. Перейдите в окно ResourceView. Раскройте элемент дерева Dialog. Выделите элемент IDD_PHONE_SDI_FORM. Для установки русского языка, находясь на этом элементе, в контекстном меню выберите Properties – Language : Russian (рисунок 141).

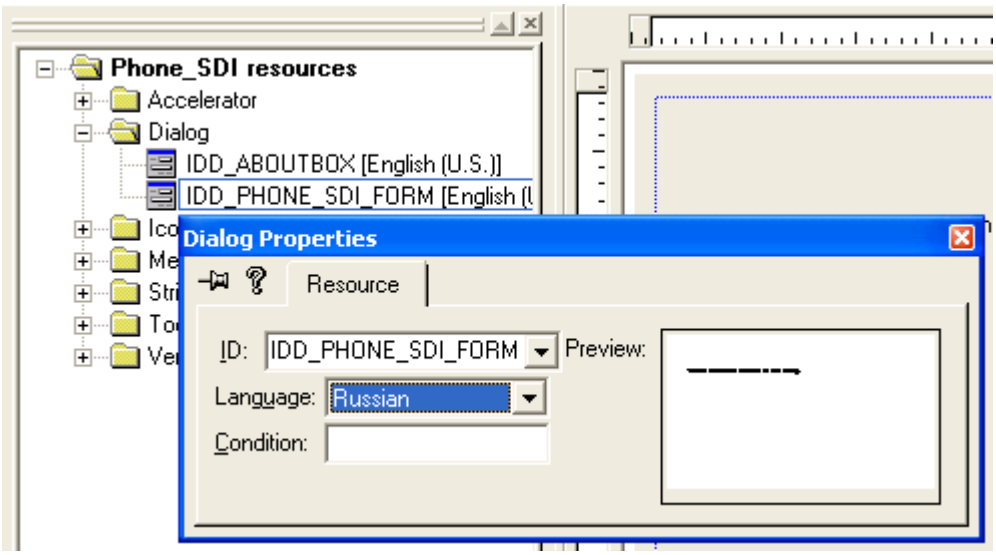



Рисунок 141. Установка русского языка

1. Сохраните изменения с помощью команды File – Save.

2. С помощью панели инструментов Contols вставьте на панель формы элементы управления в соответствии с рисунком 142 (если панель инструментов Contols не отображается, выполните команду Tools – Customize, в окне Toolbars выберите Contols).

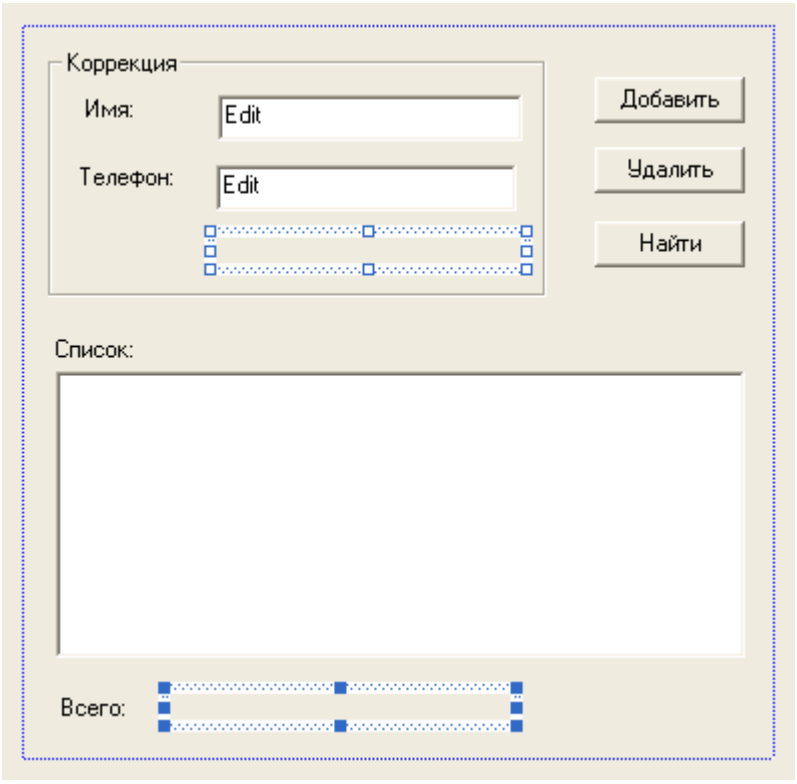



Рисунок 142. Формирование формы приложения

3. Элементам управления присвойте идентификаторы:

| Тип элемента | Заголовок | Идентификатор |
|---|---|---|
| Edit Box | | IDC_NAME |
| Edit Box | | IDC_PHONE |
| List Box | | IDC_LIST |

| | | |
|---|---|---|
| Static Text<br>(элемент управления под окном редактирования IDC_PHONE для вывода сообщения «Не найден») | | IDC_NFND |
| Static Text<br>(элемент управления для вывода общего количества записей в справочнике) | | IDC_NUM |
| Button | Добавить | IDC_ADD |
| Button | Удалить | IDC_DEL |
| Button | Найти | IDC_FIND |

4. Введите в класс CPhone_SDIView переменные, позволяющие управлять содержимым окон редактирования и окна-списка. Для этого:

- выполните команду View – ClassWizard…;
- выберите опцию Select an existing class;
- укажите имя класса CPhone_SDIView, нажмите на кнопку Select;
- выберите вкладку Member Variables;
- выберите идентификатор IDC_NAME и нажмите на кнопку Add Variable;
- заполните поля диалога в соответствии с таблицей:

| Поле | Идентификатор переменной |
|---|---|
| Member Variable name: | sName |
| Category: | Value |
| Variable type: | CString |

- повторите эти действия для поля редактирования IDC_PHONE, задав имя переменной sPhone;
- повторите эти действия для поля редактирования IDC_NFND, задав имя переменной sNotFound;
- повторите эти действия для поля редактирования IDC_NUM, задав имя переменной sTotal;
- повторите эти действия для окна-списка IDC_LIST, задав параметры в соответствии с таблицей:

| Поле | Идентификатор переменной |
|---|---|
| Member Variable name: | cList |
| Category: | Control |
| Variable type: | CListBox |

Список переменных будет следующим (рисунок 143).

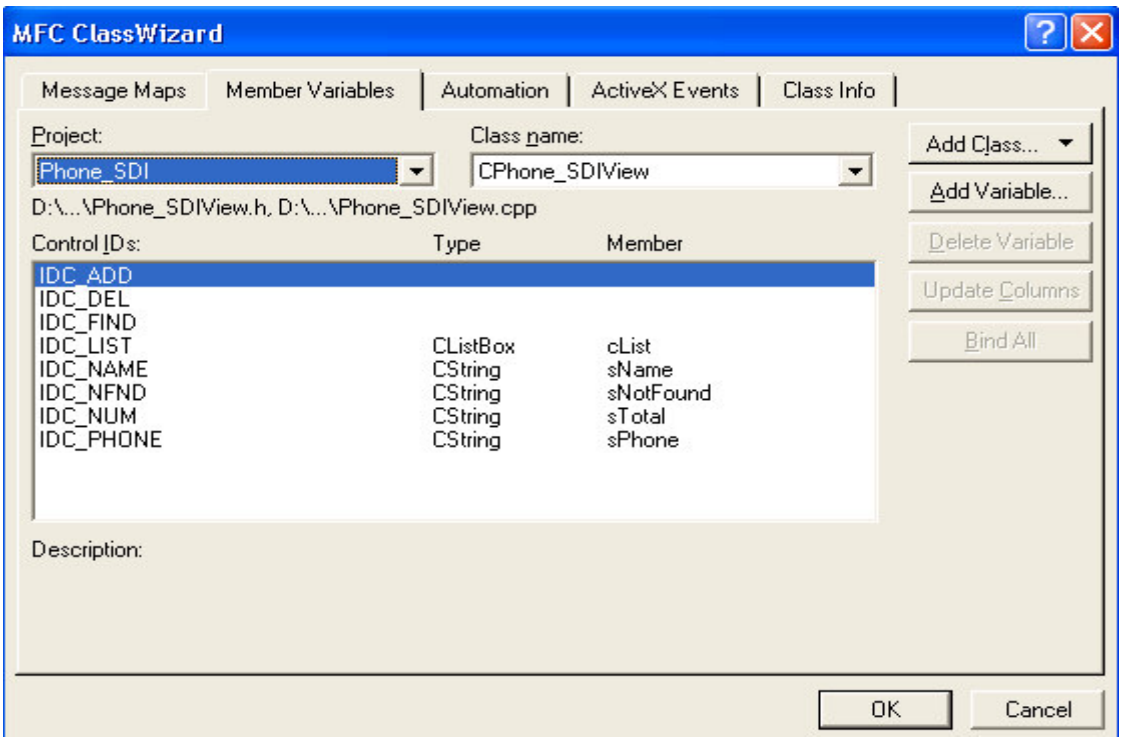



Рисунок 143. Список переменных приложения

5. Задайте функции, которые будут обрабатывать сообщения (рисунок 144), возникающие при нажатии кнопок: IDC_ADD (Добавить), IDC_DEL (Удалить) и IDC_FIND (Найти). Для этого:

- выполните двойной щелчок на кнопке Добавить;
- согласитесь с предложенным именем (OnAdd) для функции обработчика;
- повторите эти действия для двух других кнопок.

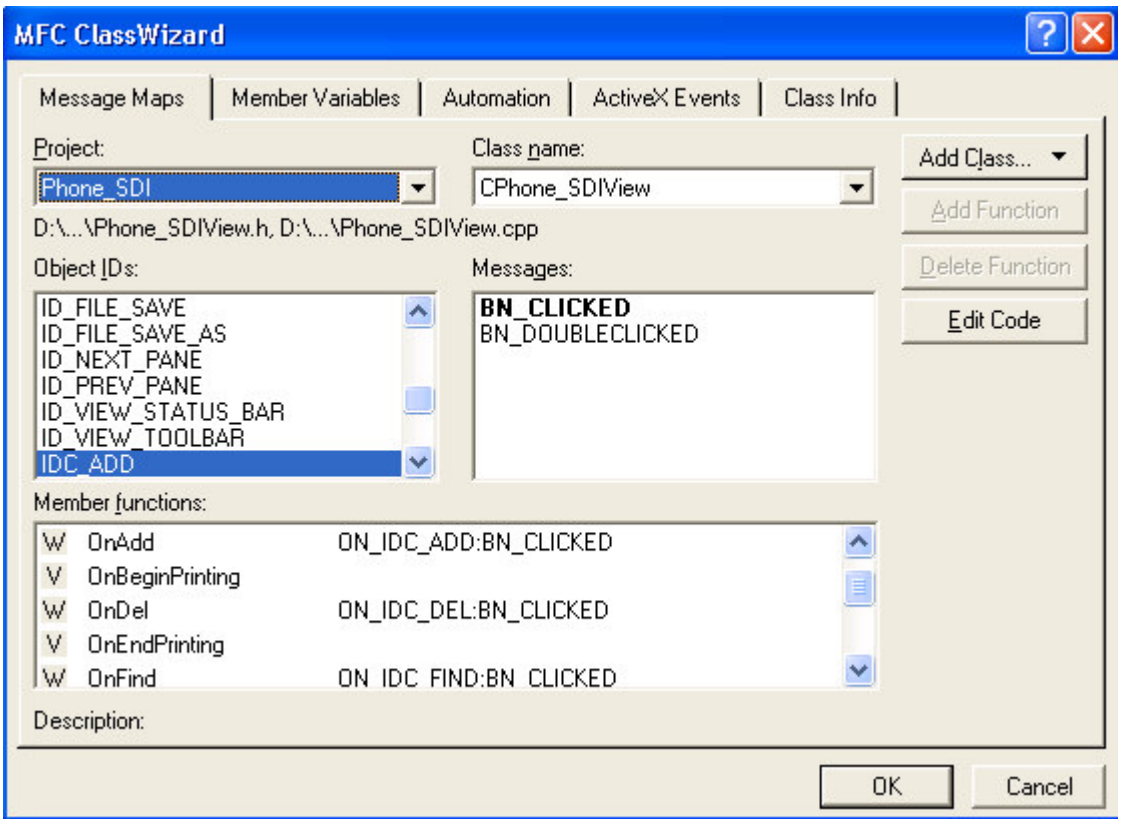


Рисунок 144. Список обработчиков сообщений приложения

6. Чтобы упростить взаимодействие классов *Вид* и *Документ,* для SDI-лриложений удобно ввести в класс *Вид* переменную CPhone_SDIDoc* pDoc; – указатель на *Документ* (но уже нужного типа), инициализировать ее один раз (в методе OnInitialUpdate) и использовать многократно, уже не обращаясь каждый раз к методу GetDocument. Кроме того, в классе *Документ* удобно объявить класс *Вид с* атрибутом friend. При этом все данные класса CPhone_SDIDoc становятся доступными внутри класса CPhone_SDIView, что также упрощает взаимодействие классов:

1) в класс CPhone_SDIDoc добавьте строку

friend class CPhone_SDIView;

2) в класс CPhone_SDIView добавьте protected-переменную

CPhone_SDIDoc* pDoc;

7. Для хранения справочника будем использовать динамическую структуру Map (Map – коллекция объектов, в которой ключевой объект ассоциирован с искомым объектом любого типа). Map – один из MFC-классов типа Collection. Для того чтобы сделать доступной библиотеку Collection, необходимо подключить заголовочный файл afxcoll.h. Вставьте директиву:

#include <afxcoll.h>

в файл StdAfx.h до строки

#ifndef _AFX_NO_AFXCMN_SUPPORT

8. В окне ClassView раскройте контекстное меню класса CPhone_SDIDoc. Выберите Add Member Variable… и задайте переменную CMapStringToString cMap; установив ей атрибут protected (рисунок 145).

Повторите эту же операцию и задайте переменную UINT nTotal; с тем же типом доступа.

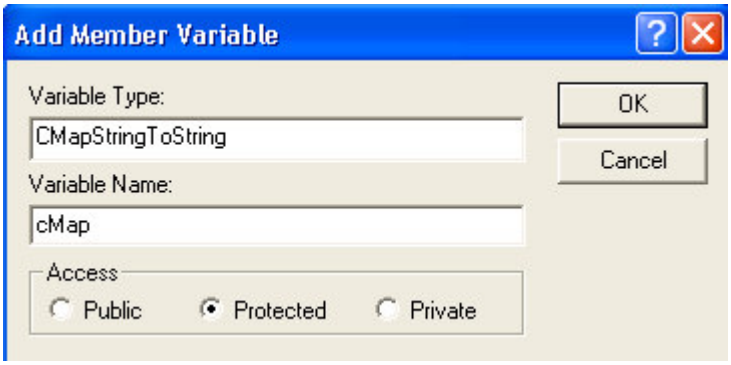



Рисунок 145. Установка атрибутов переменной

9. Внесите изменения в функцию Serialize класса CPhone_SDIDoc:

```
// CPhone_SDIDoc serialization
void CPhone_SDIDoc::Serialize(CArchive& ar)
{
   //Класс Словарь сохраняет и восстанавливает свое состояние самостоятельно
   cMap.Serialize(ar);
   if (ar.IsStoring()) //Используется при записи в архив простых переменных
   {
            ar<<nTotal;                  //Операция вывода
   }
   else
   {
            ar>>nTotal;                  //Операция ввода
```

```
    }
}
```

Вызова cMap.Serialize(ar); достаточно для выполнения операция записи и чтения содержимого справочника. Количество записей в прочитанном из файла справочнике можно узнать с помощью метода GetCount. С целью иллюстрации обмена с архивом простых переменных добавлено чтение и запись в архив (файл) общего количества элементов справочника (nTotal).

10. Задайте начальное состояние *Вида* при его открытии. Это следует делать внутри функции void CPhone_SDIView::OnInitialUpdate():

```
void CPhone_SDIView::OnInitialUpdate()
{
    //Вызов родительской версии, в которой вызывается OnUpdate
    CFormView::OnInitialUpdate();
    //Удаляем все элементы из списка на экране
    cList.ResetContent();
    pDoc = GetDocument();   //Адрес документа
    //Становимся на начало словаря
    POSITION p = pDoc->cMap.GetStartPosition();
    while (p)           //Пока в нем есть элементы, выбираем их
    {
            pDoc->cMap.GetNextAssoc(p,sName,sPhone);
            cList.AddString(MakeEntry());      //и помещаем в ListBox
    }
    cList.SetCurSel(0);                 //Выделяем первый элемент
    //Передаем его содержимое в окна редактирования
    OnDblclkList();
    GetParentFrame()->RecalcLayout();
    ResizeParentToFit();
}
```

11. Добавим функцию MakeEntry(), которая формирует запись (строку, состоящую из имени и телефона) для помещения ее в ListBox. В окне ClassView раскройте контекстное меню класса CPhone_SDIView. Выберите Add Member Function… и задайте protected-функцию MakeEntry() типа CString (рисунок 146).

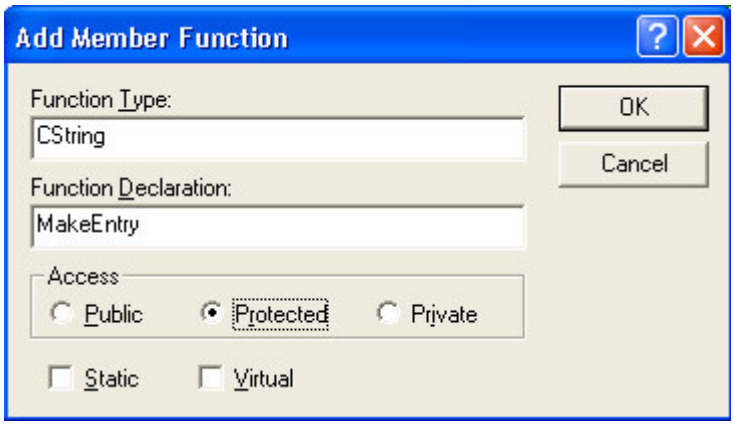



Рисунок 146. Установка атрибутов переменной

Отредактируйте код функции:

```
CString CPhone_SDIView::MakeEntry()
{
```

```
    sName.TrimLeft();
    sName.TrimRight();
    sPhone.TrimLeft();
    sPhone.TrimRight();
    return sName + ":" + sPhone;
}
```

12. Добавим функцию OnDblclkList, которая передает данные из текущей выбранной строки списка в окна редактирования:

- выполните команду View – ClassWizard…;
- выберите вкладку Message Maps;
- выберите идентификатор элемента (Object IDs:) IDC_LIST;
- в списке Message: выберите сообщение LBN_DBLCLK и нажмите на кнопку Add Function;
- согласитесь с предложенным именем (OnDblclkList) для функции обработчика;
- нажмите на кнопку OK.

Введите коды метода CPhoneDlg::OnDblclkList (), сгенерированного ClassWizard:

```
void CPhone_SDIView::OnDblclkList()
{
    int i = cList.GetCurSel();              //Выделенный элемент
    if (i == LB_ERR)                        //Если ни один не выделен, то выход
            return;
    CString s;
    cList.GetText(i,s);                     //Выбираем элемент в строку s
    //Поиск разделителя и выделение имени
    sName = s.Left(s.Find(':'));
    sPhone = s.Mid(s.Find(':')+1);
    pDoc->cMap.Lookup(sName,sPhone);        //Поиск в словаре
    UpdateTexts();
    UpdateData(FALSE);                      //Помещаем данные в форму
}
```

13. Добавьте в класс CPhone_SDIView функцию UpdateTexts(), которая убирает сообщение о том, что при поиске элемент не найден, а также реагирует на изменения общего количества элементов в справочнике, отражая эти изменения в окне на форме, связанном с переменной sTotal:

```
void CPhone_SDIView::UpdateTexts()
{
    sNotFound = "";             //Убираем сообщение
    //Определяем количество записей в справочникен
    pDoc->nTotal = pDoc->cMap.GetCount();
    //Преобразуем в строку
    sTotal.Format("%d",pDoc->nTotal);
    UpdateData(FALSE);      //Помещаем в окно
}
```

14. Введите коды метода CPhone_SDIView::OnAdd(), сгенерированного ClassWizard:

```
void CPhone_SDIView::OnAdd()
{
    UpdateData();               //Считывание полей формы
    if (sName.IsEmpty())                //Если поле имени пусто, выходим
            return;
    //Поиск в словаре и удаление на экране
```

```
    if (pDoc->cMap.Lookup(sName,sPhone))
    {
            //Если абонент существует, то поиск и удаление его из списка
            cList.DeleteString(cList.FindStringExact(-1,MakeEntry()));
            //Так как словарь восстановил новый номер телефона,
            //устанавливаем новый номер
            UpdateData();
    }
    //Добавление строки текста в список на экране
    int i = cList.AddString(MakeEntry());
    cList.SetTopIndex(i);           //Обеспечение видимости
    cList.SetCurSel(i);                     //Выделение строки
    pDoc->cMap[sName] = sPhone;     //Запись в словарь
    UpdateTexts();                  //Корректируем количество записей
    //Помечаем документ как измененный (требующий сохранения)
    pDoc->SetModifiedFlag();
    sName = "";
    sPhone = "";
    UpdateData(FALSE);
}
```

15. Введите коды метода void CPhone_SDIView::OnDel(), сгенерированного ClassWizard:

```
void CPhone_SDIView::OnDel()
{
    CString s;
    int i = cList.GetCurSel();          //Выделенный элемент
    if (i == LB_ERR)                    //Если нет, то выходим
            return;
    cList.GetText(i,s);                 //Выбираем строку
    cList.DeleteString(i);              //Удаление из списка
    //Поиск разделителя и выделение имени
    sName = s.Left(s.Find(':'));
    pDoc->cMap.RemoveKey(sName);            //Удаление из словаря
    int last = cList.GetCount()-1;      //Индекс последней строки
    i = i <= last ? i : i-1;                //Поиск соседнего элемента
    cList.SetCurSel(i);                     //Выделяем i-ю строку
    UpdateTexts();                      //Корректируем сумму
    pDoc->SetModifiedFlag();                //Документ изменен
    sName = "";
    sPhone = "";
    UpdateData(FALSE);
}
```

16. Сохраните приложение. Проверьте работу приложения (рисунок 147).

17. Введите коды метода void CPhone_SDIView::OnDel(), сгенерированного ClassWizard:

```
void CPhone_SDIView::OnFind()
{
    int i = -1;                         //Сначала не найден
    UpdateData();                   //Выбор имени для поиска
    if (sName.IsEmpty())                //Если нет, выходим
            return;
    sName.TrimLeft();                   //Удаление пробелов
```

```
    sName.TrimRight();
    sNotFound = "";                    //Очистка строки сообщения
    //Поиск в словаре, и если не найден
    if (pDoc->cMap.Lookup(sName,sPhone))
                                                        //то ищем в списке на экране
            i = cList.FindStringExact(-1,MakeEntry());
    else
            //Если не найден, то ищем приблизительно
            if ((i = cList.SelectString(-1,sName)) == LB_ERR)
                    sNotFound = "Не найден";            //Если опять не найден,
//сообщаем о неудаче
    if (i>=0)           //Если найден, обеспечиваем видимость на экране
    {
            cList.SetTopIndex(i>0 ? i-1 : i);
            cList.SetCurSel(i);
    }
    UpdateData(FALSE);                 //Передаем в поля для редактирования
}
```

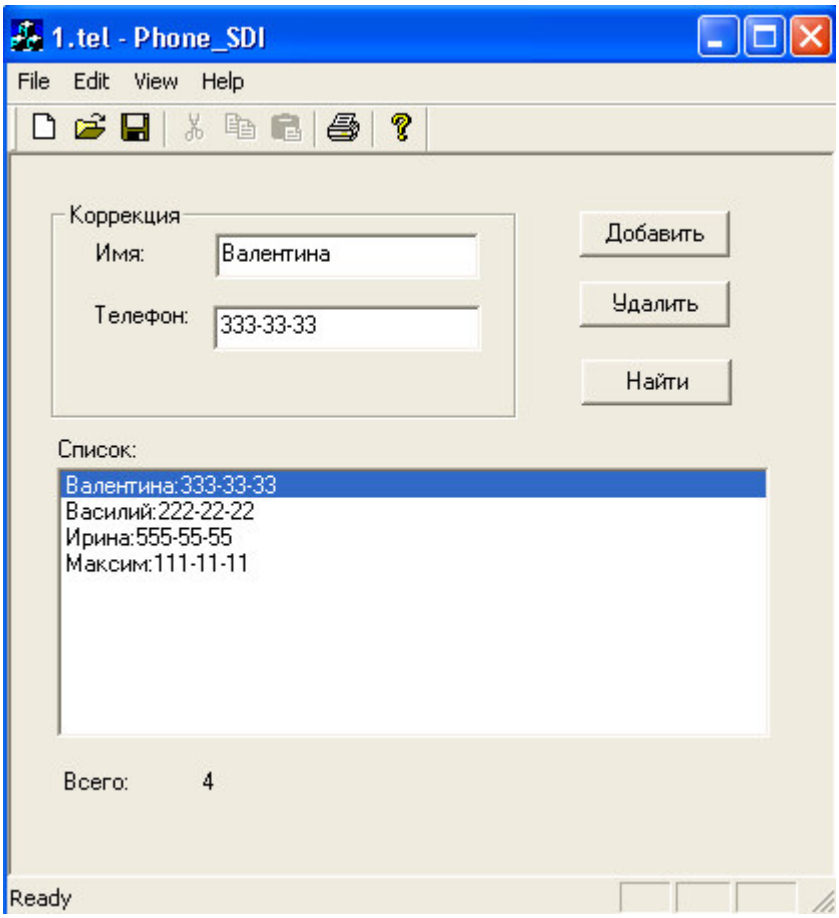



Рисунок 147. Проверка работы приложения

18. Сохраните приложение. Проверьте работу приложения.

19. Для того чтобы при изменении *Документа* изменялся *Вид*, необходимо внести следующие изменения в программу:

- в класс CPhone_SDIDoc с помощью ClassWizard добавьте и отредактируйте функцию DeleteContents():

```
void CPhone_SDIDoc::DeleteContents()
{
    cMap.RemoveAll();                    //Удаление словаря
    CDocument::DeleteContents();
}
```

- внесите изменение в функцию CPhone_SDIDoc::OnNewDocument():

```
BOOL CPhone_SDIDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;
    nTotal = 0;
    UpdateAllViews(NULL);        //Уведомление Вида
    return TRUE;
}
```

- в класс CPhone_SDIView с помощью ClassWizard добавьте и отредактируйте виртуальную функцию OnUpdate:

```
void CPhone_SDIView::OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)
{
    sName = "";                          //Очищаем поля
    sPhone = "";
    sNotFound = "";
    sTotal = "";
    UpdateData(FALSE);
}
```

Сохраните приложение. Проверьте работу приложения. При нажатии кнопки New (или выборе пункта меню File – New) *Вид* действительно обновляется, то же происходит при открытии другого Документа (файла).

20. Для того чтобы при запуске приложения по умолчанию открывался файл, содержащий существующий телефонный справочник, необходимо внести следующие изменения в программу:

- в класс CPhone_SDIDoc добавьте переменную:

```
bool bFirst;
```

- инициализируйте переменную bFirst в конструкторе класса CPhone_SDIDoc:

```
bFirst = TRUE;
```

- измените значение переменной bFirst после первой инициализации окна. Для этого в конце метода OnInitialUpdate() добавьте оператор:

```
pDoc->bFirst = FALSE;
```

- внесите изменения в метод OnNewDocument():

```
BOOL CPhone_SDIDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;
    if (!bFirst)          //Если это не первоначальный вызов, то выходим
        return TRUE;
    CFile file;                  //Класс, управляющий файлом
    CFileException e;            //Объект класса для обработки сбоев
    CString fn = "d:\\MyPhones.tel";        //Имя файла
    if (!file.Open(fn,CFile::modeReadWrite,&e))
```

```
{
        //Если не удалось открыть файл, объясняем причину
        if (e.m_cause == CFileException::fileNotFound)
                AfxMessageBox("Файл " + fn + " не найден");
        else
                AfxMessageBox("Ошибка открытия файла " + fn);
        return TRUE;
}
//Связываем архив с файлом
CArchive ar(&file,CArchive::load);
Serialize(ar);                  //Чтение из файла
SetPathName(fn);                //Отображаем имя файла в заголовке
UpdateAllViews(NULL);           //Уведомление Вида
return TRUE;
}
```

Сохраните приложение. Проверьте работу приложения.

21. Чтобы при запуске приложения автоматически открывался документ, который был открыт в момент завершения последней сессии, замените строку

```
CString fn = "d:\\MyPhones.tel";
```

на

```
CString fn = AfxGetApp()->GetProfileString("Recent File List", "File1");
```

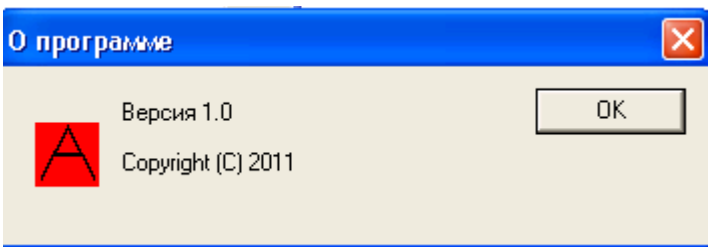
Имя первого файла (File1) из списка последних открывавшихся файлов берется из Windows-реестра с помощью метода GetProfileString.

***Самостоятельная работа***

*Вариант 1*

Внесите в созданное приложение следующие изменения.

1. Измените диалоговое окно About Phone_SDI в соответствии с рисунком:




2. Удалите из меню пункты: File – Print…; File – Print Preview, File – Print Setup… Русифицируйте меню.

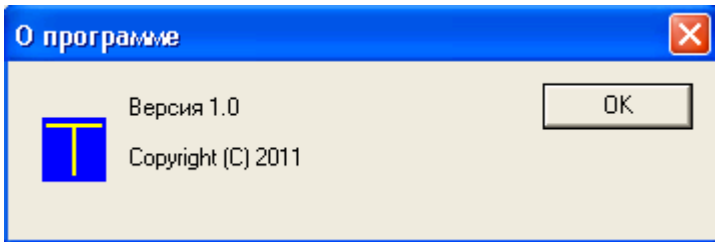
3. Измените функциональную клавишу, используемую для открытия документа, на Ctrl+Z.

4. Русифицируйте сообщения, выводимые в строке состояния при наведении курсора мыши на пункты меню.

*Вариант 2*

Внесите в созданное приложение следующие изменения.

1. Измените диалоговое окно About Phone_SDI в соответствии с рисунком:




2. Удалите из меню пункты: File – Print…; File – Print Preview, File – Print Setup… Русифицируйте меню.

3. Смените функциональную клавишу, используемую для сохранения документа, на Ctrl+Z.

4. Русифицируйте сообщения, выводимые в строке состояния при наведении курсора мыши на пункты меню.

*Вариант 3*

Внесите в созданное приложение следующие изменения.

1. Измените диалоговое окно About Phone_SDI в соответствии с рисунком:


2. Удалите из меню пункты: File – Print…; File – Print Preview, File – Print Setup… Русифицируйте меню.

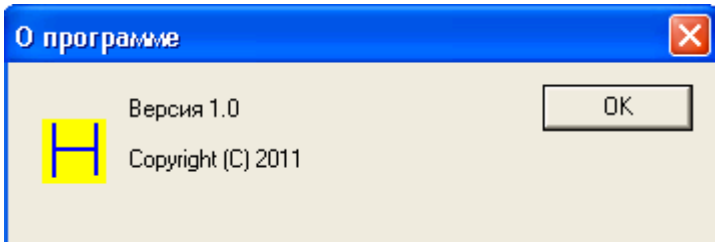
3. Смените функциональную клавишу, используемую для создания документа, на Ctrl+Z.

4. Русифицируйте сообщения, выводимые в строке состояния при наведении курсора мыши на пункты меню.

*Вариант 4*

Внесите в созданное приложение следующие изменения.

1. Измените диалоговое окно About Phone_SDI в соответствии с рисунком:




2. Удалите из меню пункты: File – Print…; File – Print Preview, File – Print Setup… Русифицируйте меню.

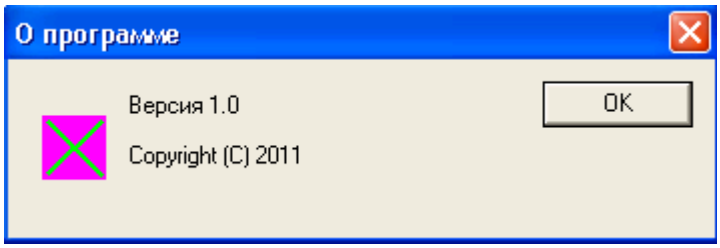
3. Измените функциональную клавишу, используемую для печати документа, на Ctrl+Z.

4. Русифицируйте сообщения, выводимые в строке состояния при наведении курсора мыши на кнопки панели инструментов.

*Вариант 5*

Внесите в созданное приложение следующие изменения.

1. Измените диалоговое окно About Phone_SDI в соответствии с рисунком:




2. Удалите из меню пункты: File – Print…; File – Print Preview, File – Print Setup… Русифицируйте меню.

3. Измените функциональную клавишу, используемую для копирования информации в буфер обмена, на Ctrl+Z.

4. Русифицируйте сообщения, выводимые в строке состояния при наведении курсора мыши на кнопки панели инструментов.

## Лабораторный практикум № 18. Разработка MDI-приложения на языке C++

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 8.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение основных принципов работы с библиотекой классов MFC.

***Вводная часть***

*Особенности MDI-приложения*

Если приложение может одновременно использовать только один документ, то говорят, что оно имеет однодокументный интерфейс (SDI – Single Document Interface). Если оно одновременно может открыть несколько документов, то говорят, что приложение имеет многодокументный интерфейс (MDI – Multiple Document Interface). Классическим примером SDI-приложения является редактор Notepad, а MDI-приложения – MS Word.

MDI-приложение отображает информацию как в главном окне приложения, так и в дочерних окнах. Главное обрамляющее окно является потомком класса CMDIFrameWnd, а дочерние окна - класса CMDIChildWnd. Главное обрамляющее окно MDI-приложения содержит специальное дочернее окно, называемое MDICLIENT-окно. Это последнее окно управляет клиентской областью главного обрамляющего окна и само имеет дочерние окна: окна документов, производные от CMDIChildWnd. Особенностью MDICLIENT-окна является то, что оно остаётся невидимым.

Рисунок 148 показывает взаимосвязь между MDI-обрамляющим окном, его окном MDICLIENT и дочерними окнами документов.

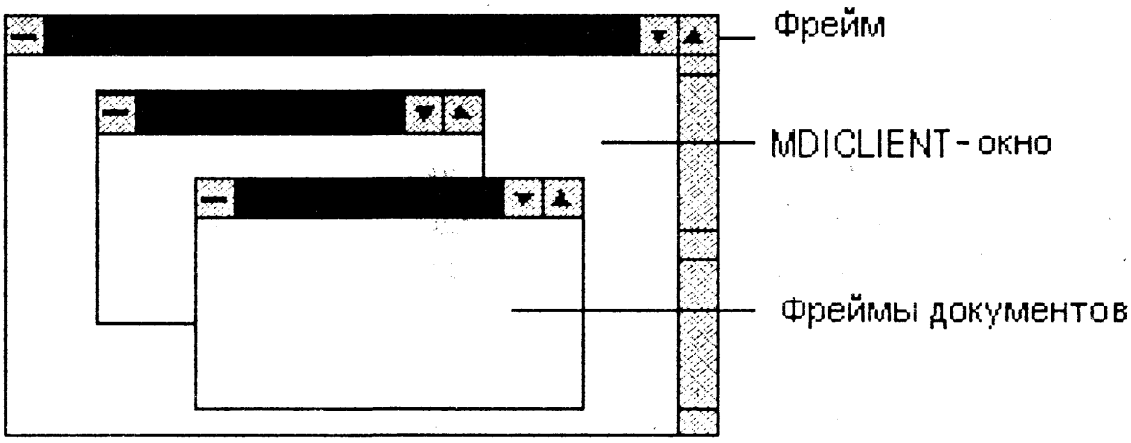



Рисунок 148. Взаимосвязь между MDI-обрамляющим окном, его окном MDICLIENT и дочерними окнами документов

*Роль фрейма в архитектуре «документ/представление»*

Фрейм документа имеет две основные составляющие – собственно фрейм и его содержимое. Эти два компонента представлены и управляются различными группами классов библиотеки MFC: классы фреймов (CFrameWnd CMDIChildWnd) и классы представлений (CView, CScrollView и др.). Взаимосвязь между фреймом и представлением показана на рисунке 149.

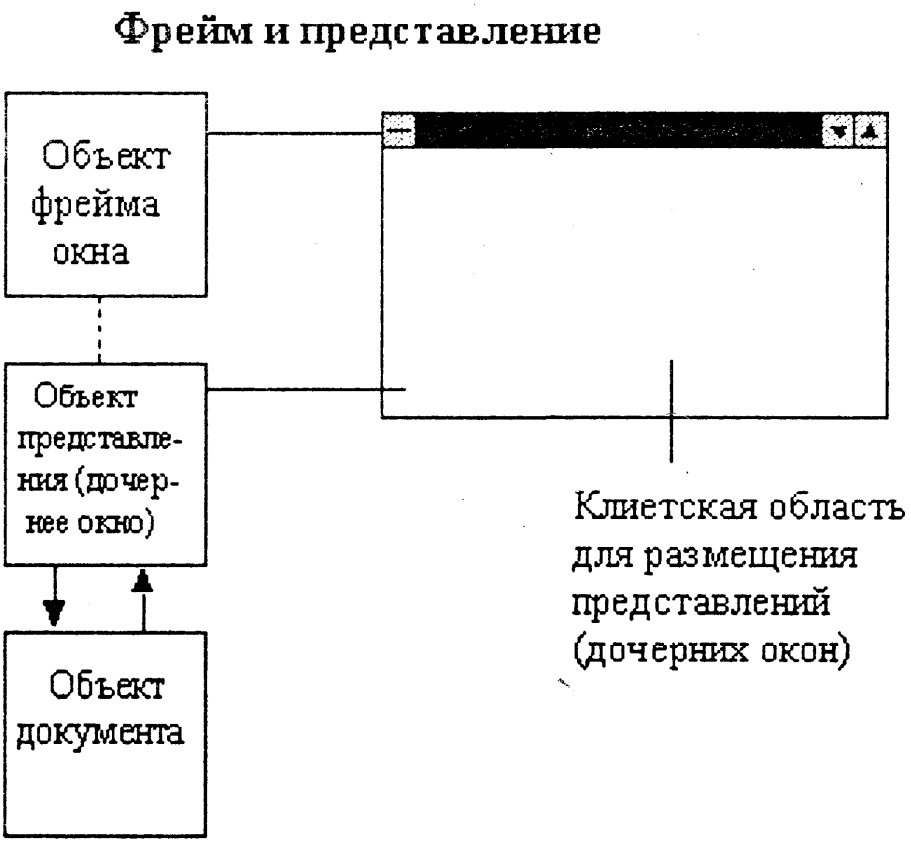



Рисунок 149. Взаимосвязь между фреймом и представлением

Фрейм хранит путь к текущему активному представлению. Если он содержит более одного представления, например, при работе с разделённым окном, то в качестве активного принимается последнее используемое представление. Фрейм имеет доступ к документу, ассоциированному с текущим активным представлением. Кроме того, фрейм занимается координацией взаимодействия межу различными представлениями, ассоциированными с определённым документом, направляя их команды и получая от них извещения.

Фрейм также решает задачу разделения ресурсов среди документов, созданных на основе одного шаблона, и замены ресурсов при активизации документа другого типа.

CView* GetActiveView(); – функция возвращает указатель на объект текущего активного представления. Иначе NULL.

Void SetActiveView(CView* pViewNew, BOOL bNotify = TRUE); – функция переводит представление pViewNew в активное стояние.

Virtual CFrameWnd* GetActiveFrame(); – функция возвращает указатель на активное MDI-дочернее окно (класса CMDIChildWnd). Если приложение не имеет активного MDI-дочернего окна или приложение имеет SDI-интерфейс, возвращается неявный указатель this.

Virtual CDocument* GetActiveDocument(); – функция возвращает указатель на текущий документ. Если текущего документа нет, возвращает NULL.

Ниже перечислены шаги, которые необходимо проделать для организации работы в рамках архитектуры «документ/представление»:

• создать объект-приложение;

• создать объекты-документы;

• создать нужное число представлений каждого документа;

• в функции InitInstance() объекта-приложения создать объекты-шаблоны документов и добавить их к его списку;

• создать необходимые ресурсы для каждого документа;

• переопределить функцию CFrameWnd::OnCreateClient(), в которой присоединить к фрейму документа необходимые представления;

• создать и вывести на экран главное окно приложения с присоединёнными к нему фреймами документа.

***Практическая часть***

*Генерация MDI-приложения с помощью MFC AppWizard*

1. Запустите Microsoft Visual Studio.

2. Выполните команду File – New… Выберите вкладку Projects.

3. Выберите тип проекта MFC AppWizard(exe). В поле Location установите рабочую директорию. В поле Project name введите имя проекта (рисунок 150).

4. Нажмите на кнопку OK. После нажатия кнопки OK инструмент AppWizard будет показывать страницы диалога для уточнения проекта (шаги определения установок проекта). Сделайте выбор в соответствии с таблицей 18.

5. Visual Studio отобразит диалоговое окно New Project Information (информация о новом проекте) (рисунок 151).

Убедитесь, что все установки вашей программы корректны (при необходимости нажмите на кнопу Cancel и внесите изменения). Нажмите на кнопку OK. MFC AppWizard создаст оболочку программы.

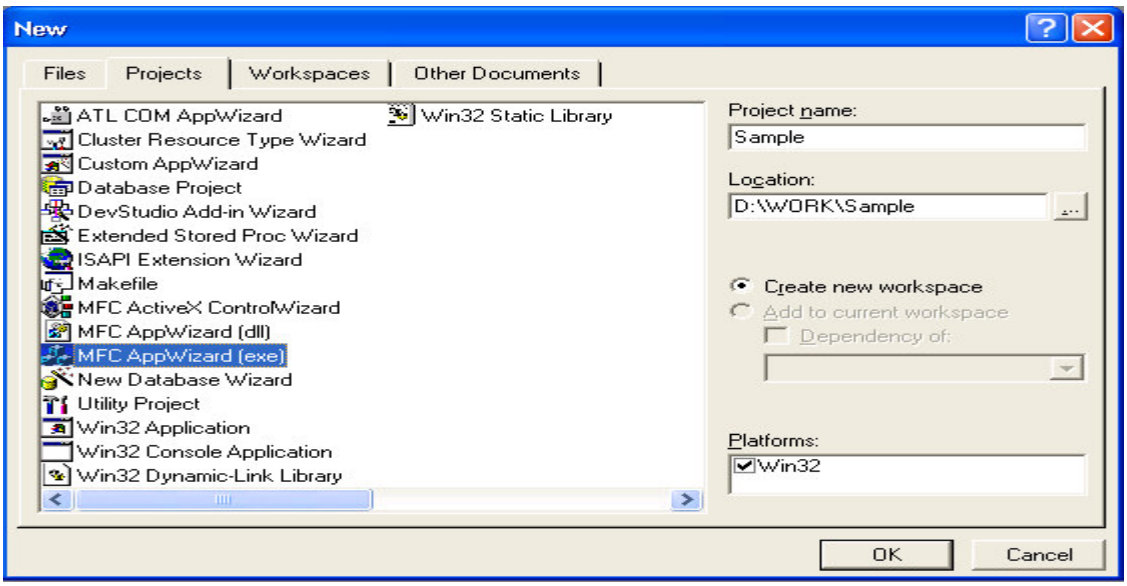
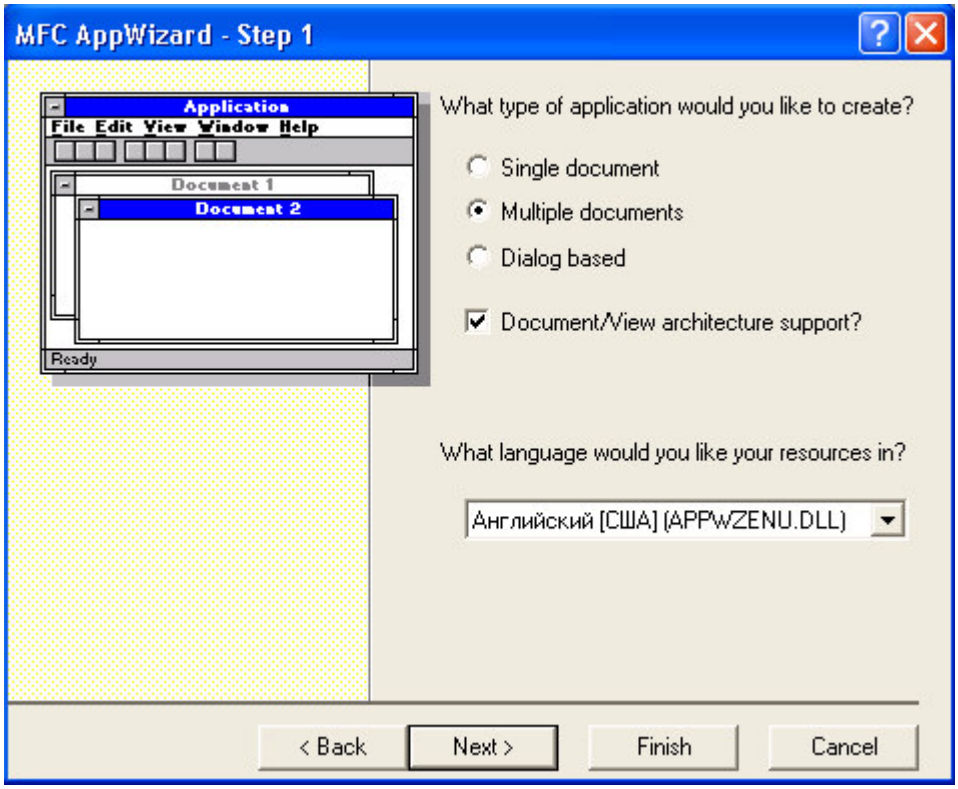
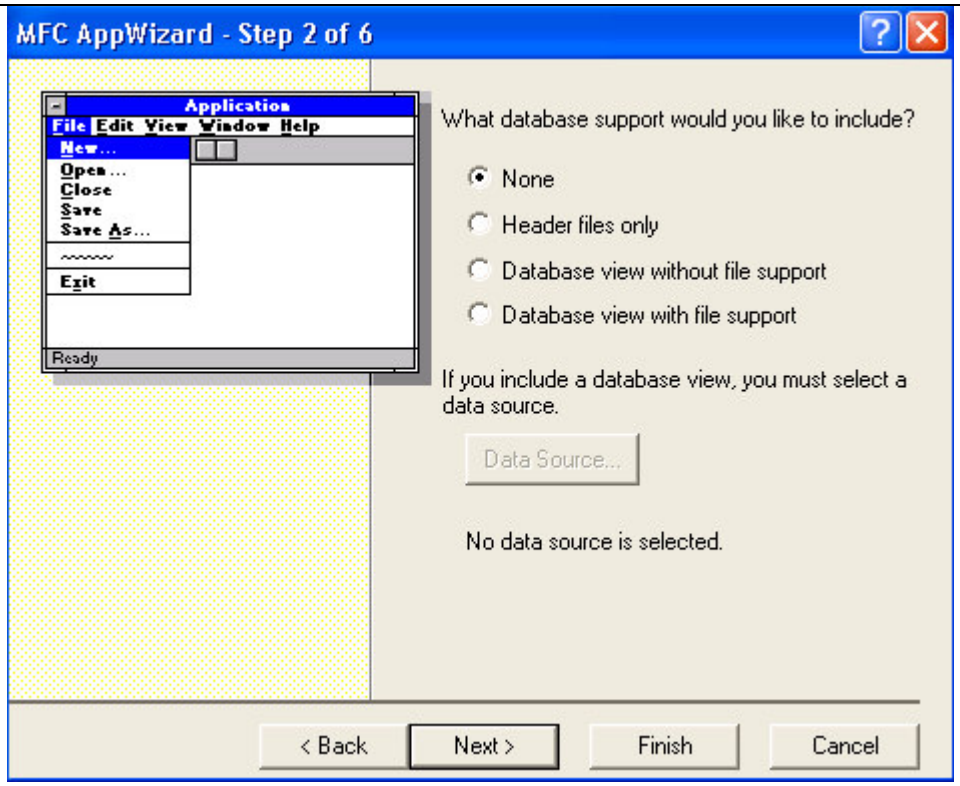
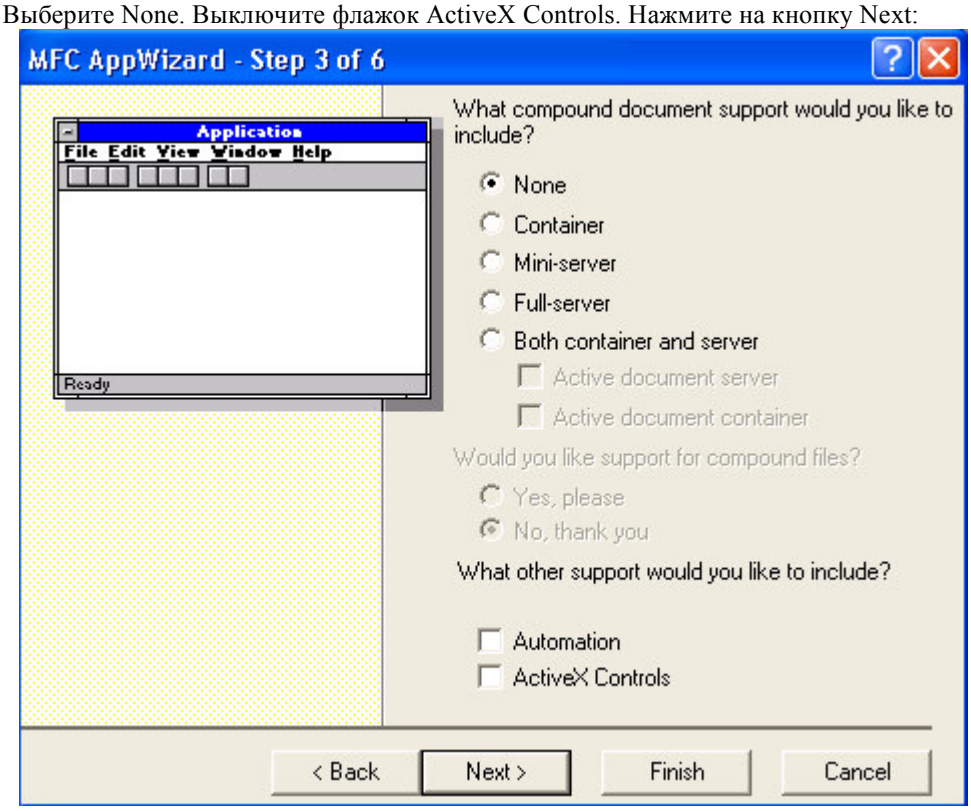
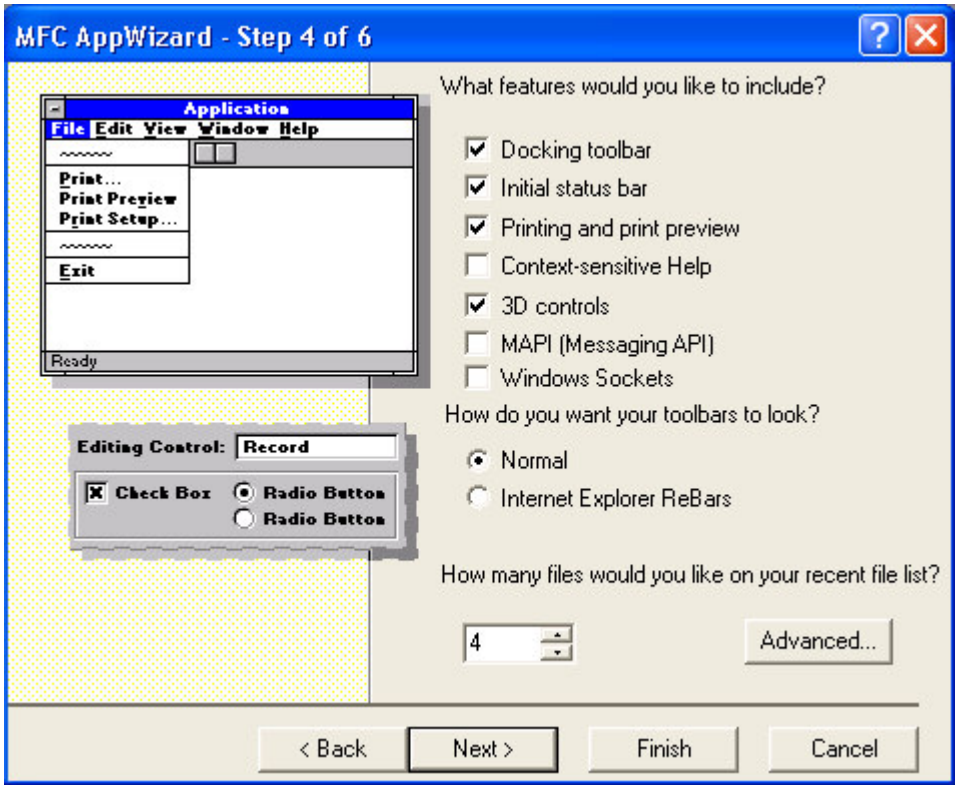
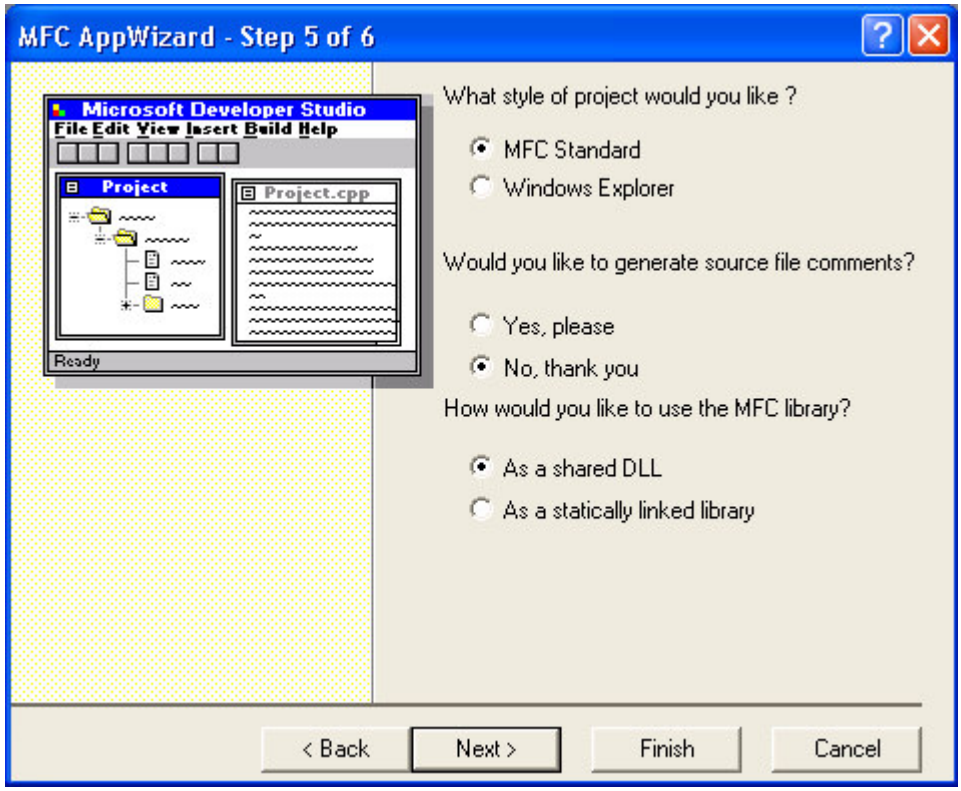



Рисунок 150. Выбор типа приложения

Таблица 18. Этапы разработки MDI-приложения

| Шаг | Выбор или действие |
|---|---|
| 1 (Выбор типа приложения) | Выберите переключатель Multiple documents (многодокументный интерфейс). Нажмите на кнопку Next:  |
| 2 (Поддержка базы данных) | Выберите None. Нажмите на кнопку Next: |

| Шаг | Выбор или действие |
|---|---|
| |  |
| 3 (Поддержка составных документов) | Выберите None. Выключите флажок ActiveX Controls. Нажмите на кнопку Next:<br>  |
| 4 (Включение свойств – стандартных возможностей, включаемых в приложение) | Установите параметры в соответствии с рисунком: |

| Шаг | Выбор или действие |
|---|---|
| |  |
| 5 (Подключение библиотеки MFC) | Выберите переключатели MFC Standard; No, thank you; As a shared Dll. Нажмите на кнопку Next:<br>  |

| Шаг | Выбор или действие |
|---|---|
| 6 (Создание классов приложения) | Нажмите на кнопку Finish: |

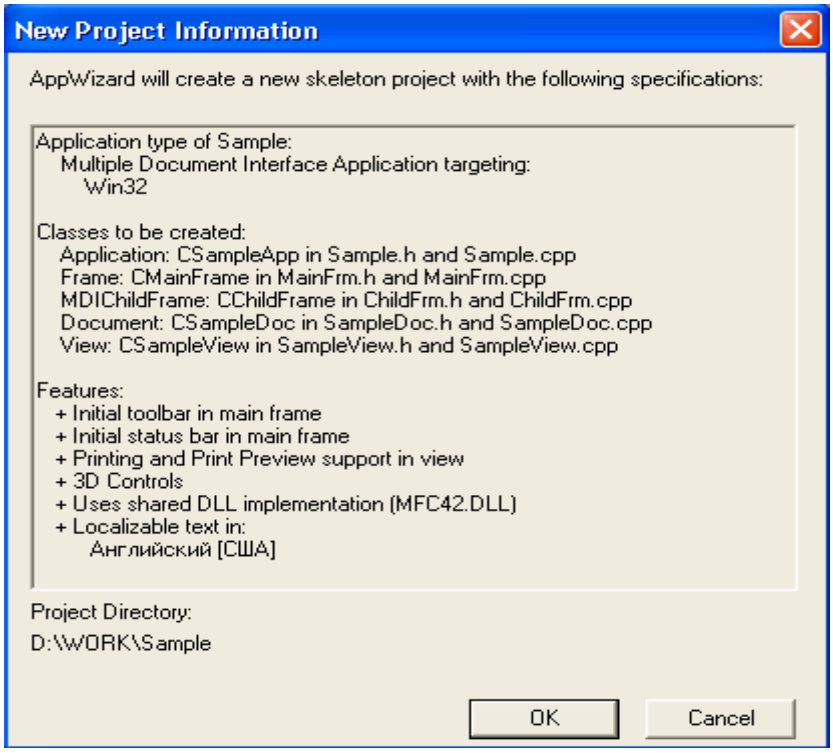



Рисунок 151. Завершение генерации приложения

Созданное приложение состоит из шести классов. Сгенерированный остов выполнен в соответствии с моделью программирования *Документ-Вид.* В MDI-приложении каждый тип документа *(Document template)* может иметь несколько *Видов,* как разных, так и одинаковых. Каждый *Вид* отображается в отдельном окне-потомке (child window) главного окна-рамки (frame window).

• CSampleApp – класс приложения, поддерживающий функциональность MDI-приложения.

• CSampleDoc – класс *Документ,* поддерживающий функциональность документов MDI-приложения. *

• CSampleView – класс *Вид,* поддерживающий один из многих возможных обликов документа.

• CMainFrame – класс главного окна приложения или окна-рамки.

• CChildFrame – класс окон-потомков. Каждое окно содержит *Вид* какого-либо документа.

• CAboutDlg – класс диалога About меню Help.

Все эти классы являются производными от классов MFC. У всех них есть общие предки – классы CObject и CCmdTarget (причем последний является потомком первого). Наследование данных и методов, поддерживаемое языком C++, означает, что все шесть классов приложения унаследовали функциональность этих классов. Например, CCmdTarget передал классам приложения способность реагировать на команды пользователя и сообщения Windows, обрабатывая их в функциях-обработчиках *(Message handlers.*

6. Запустите приложение (рисунок 152). MDI-проект отличается тем, что он поддерживает многодокументный интерфейс. Запущенное приложение имеет два вида меню: одно меню действует при отсутствии окон-потомков или видов документа, другое – при их наличии:

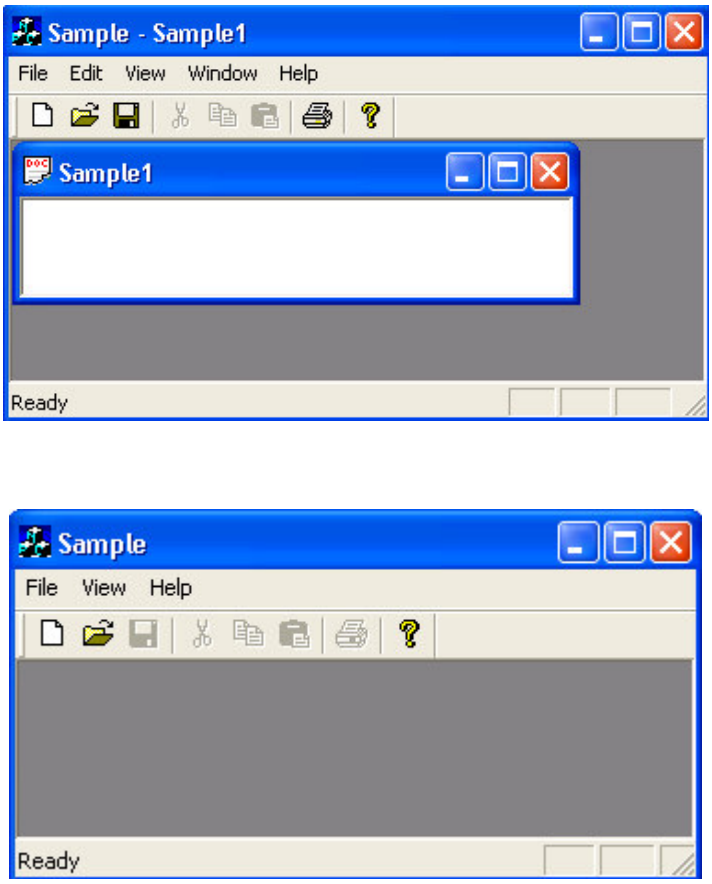



Рисунок 152. Запуск сгенерированного приложения

7. Создайте несколько новых видов (дочерних окон в клиентской области главного окна), нажимая кнопку New, или с помощью команды New меню File. Проанализируйте функциональность всех команд меню приложения. Завершите работу приложения.

*Разработка прикладной части приложения*

1. Перейдите в окно ResourceView. Раскройте элемент дерева Dialog. Выделите элемент IDD_ABOUTBOX. Для установки русского языка, находясь на этом элементе, в контекстном меню выберите Properties – Language : Russian (рисунок 153).

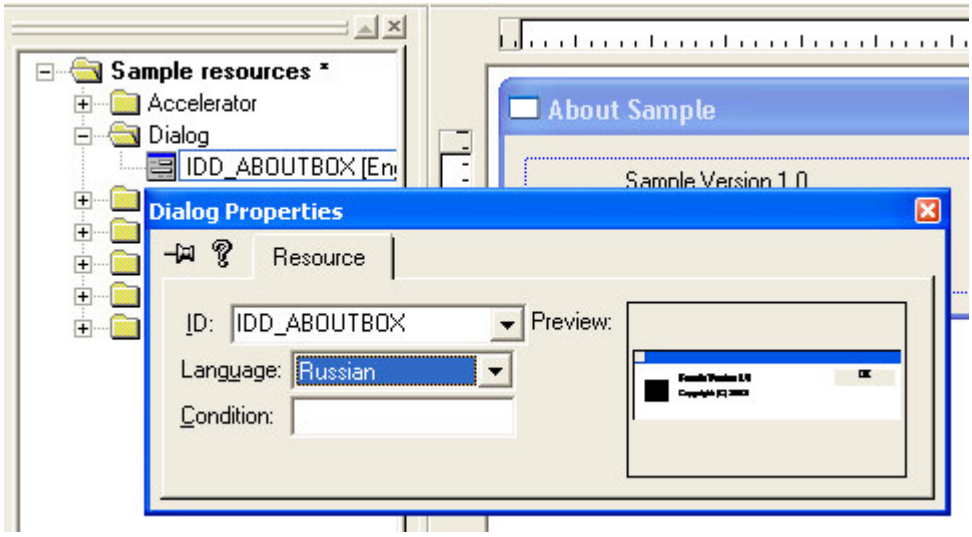



Рисунок 153. Установка русского языка

Аналогичным образом установите русский язык для ресурсов Menu и String Table приложения.

1. Сохраните изменения с помощью команды File – Save.

2. Измените заголовок главного окна приложения. Для этого откройте ресурс String Table, дважды щелкните мышью по строке с идентификатором IDR_MAINFRAME и введите в поле Caption новый заголовок (рисунок 154).

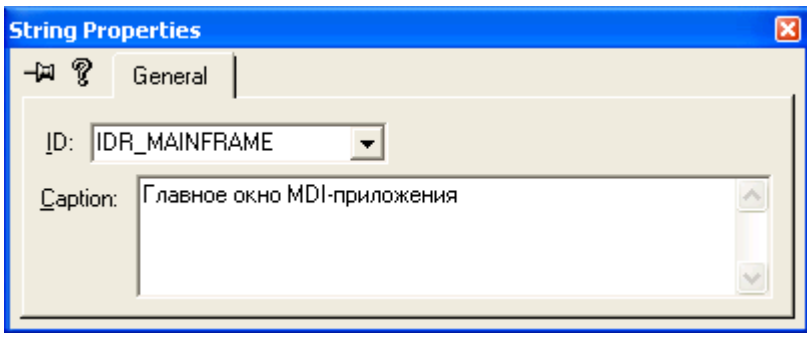



Рисунок 154. Установка заголовка приложения

3. Вставьте код программы в начале процедуры BOOL CSampleApp::InitInstance(), проверяющий, чтобы приложение не запускалось повторно:

```
CString csCaption;
// Загружаем строковый ресурс с именем приложения
csCaption.LoadString(IDR_MAINFRAME);
// Проверяем, не было ли приложение уже загружено
CWnd *pWnd = CWnd::FindWindow(NULL, csCaption);
// Если да, то делаем его видимым и переводим на передний план
                                        if (pWnd != NULL) {
        if (pWnd->IsIconic())           // Возвращает True, если окно свернуто
                pWnd->ShowWindow(SW_RESTORE);  // Восстанавливаем окно
        pWnd->SetForegroundWindow();  // Помещаем окно на передний план
        return FALSE;
}
```

4. Закомментируйте строки процедуры BOOL CSampleApp::InitInstance(), чтобы дочернее окно приложения не создавалось автоматически:

```
// Register document templates
/*
    CMultiDocTemplate* pDocTemplate;
    pDocTemplate = new CMultiDocTemplate(
            IDR_SAMPLETYPE,
            RUNTIME_CLASS(CSampleDoc),
            RUNTIME_CLASS(CChildFrame), // custom MDI child frame
            RUNTIME_CLASS(CSampleView));
    AddDocTemplate(pDocTemplate);
*/
```

а также закомментируйте строки файла Sample.cpp:

```
//  ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
//  ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
```

// ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)

5. Сохраните и проверьте работу приложения (рисунок 155).

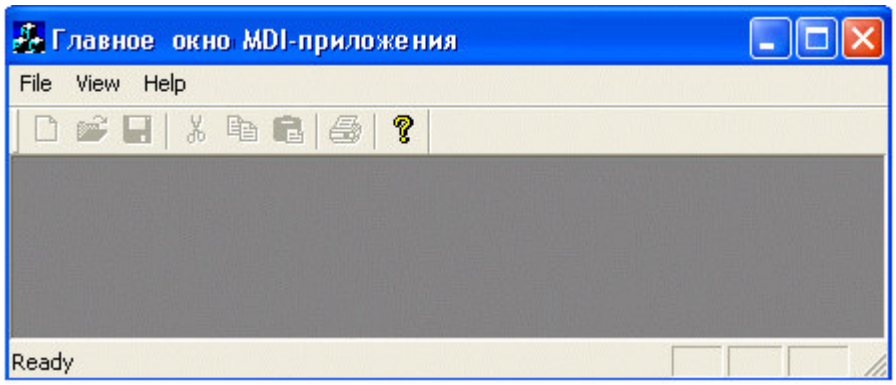



Рисунок 155. Проверка работы приложения

6. Внесите изменение в главное меню приложения IDR_MAINFRAME. Для этого откройте ресурс IDR_MAINFRAME. Вызовите контекстное меню для пункта File, выберите Properties, в поле Caption введите Файл (рисунок 156).

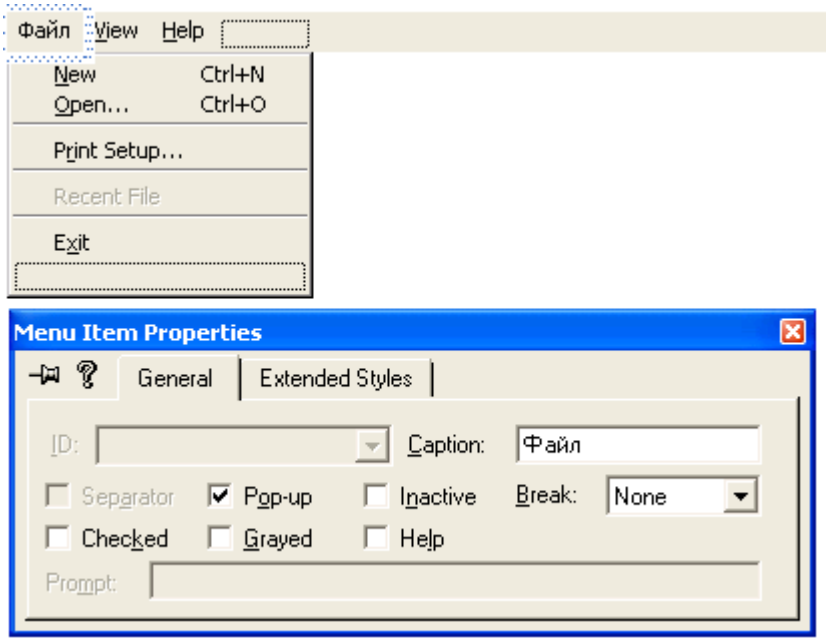



Рисунок 156. Формирование главного меню приложения

Вызовите контекстное меню для пункта New, выберите Properties, в поле Caption введите – Новое окно, измените идентификатор на ID_FILE_TEXT, в поле Prompt введите отображаемый комментарий – Новое окно (рисунок 157).

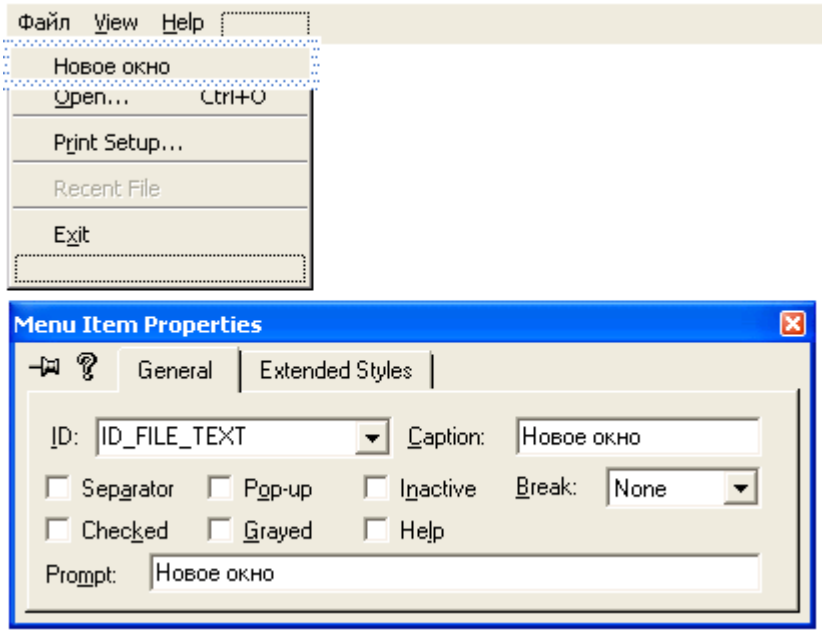



Рисунок 157. Формирование главного меню приложения

Измените главное меню приложения в соответствии с рисунком 158 и данными таблицы 19 (для удаления ненужных пунктов меню используйте клавишу Del):

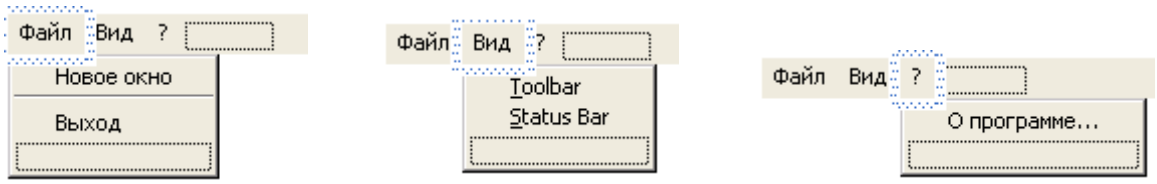



Рисунок 158. Формирование главного меню приложения

Таблица 19. Описание пунктов главного меню приложения

| Caption | ID | Prompt |
|---|---|---|
| Файл | | |
| Новое окно | ID_FILE_TEXT | Новое окно |
| Выход | ID_APP_EXIT | Выход |
| Вид | | |
| Toolbar | ID_VIEW_TOOLBAR | Скрыть/показать Toolbar |
| Status Bar | ID_VIEW_STATUS_BAR | Скрыть/показать StatusBar |
| ? | | |
| О программе… | ID_APP_ABOUT | О программе |

7. Внесите изменение в меню приложения IDR_SAMPLETYPE, вызываемого для дочерних окон. Для этого откройте ресурс IDR_SAMPLETYPE. Измените меню в соответствии с рисунком 159 и данными таблицы 20.

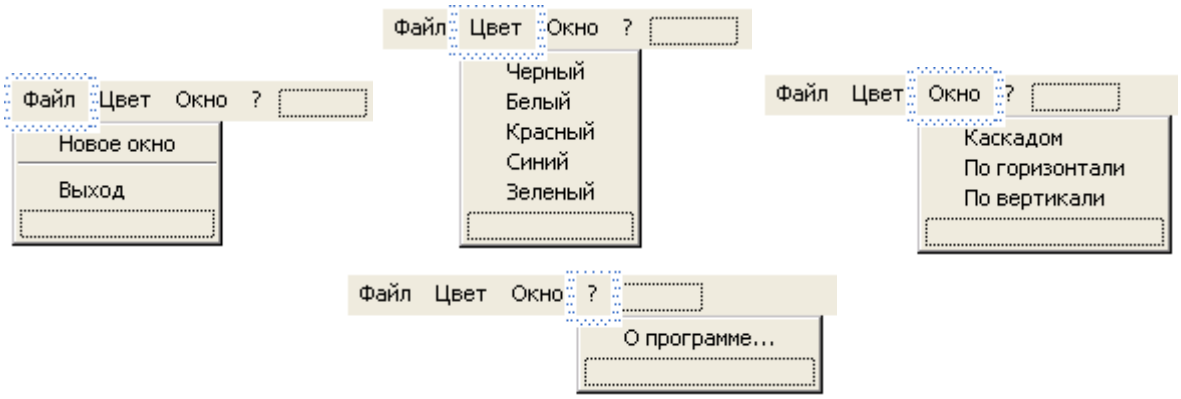



Рисунок 159. Редактирование меню дочерних окон приложения

Таблица 20. Описание пунктов меню дочерних окон приложения

| Caption | ID | Prompt |
|---|---|---|
| Файл | | |
| Новое окно | ID_FILE_TEXT | Новое окно |
| Выход | ID_APP_EXIT | Выход |
| Цвет | | |
| Черный | IDM_BLACK | Черный |
| Белый | IDM_WHITE | Белый |
| Красный | IDM_RED | Красный |
| Синий | IDM_BLUE | Синий |
| Зеленый | IDM_GREEN | Зеленый |
| Окно | | |
| Каскадом | ID_WINDOW_CASCADE | Окна каскадом |
| По горизонтали | ID_WINDOW_TILE_HORZ | Окна по горизонтали |
| По вертикали | ID_WINDOW_TILE_VERT | Окна по вертикали |
| ? | | |
| О программе… | ID_APP_ABOUT | О программе |

8. Зарегистрируйте новый класс окна Windows для дочерних окон приложения. Для этого внесите изменения в процедуру int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct) – обработчик сообщения WM_CREATE главного окна приложения, добавив в конце перед оператором return 0; следующие строки программного кода:

```
    // Регистрируем новый класс окна Windows для “MDI child”
lpszTextClass = AfxRegisterWndClass(
            CS_HREDRAW | CS_VREDRAW,
            AfxGetApp()->LoadCursor(IDC_ARROW),
            (HBRUSH) (COLOR_WINDOW+1),
            AfxGetApp()->LoadIcon(IDR_SAMPLETYPE));
    // Вызов обработчика команды для создания и вывода на экран первого
    //  дочернего окна MDI-приложения
OnFileText();
```

В начале модуля MainFrm.cpp опишите переменную:

```
LPCTSTR lpszTextClass;
```

и добавьте файл, содержащий описание класса CChildFrame:

```
#include "ChildFrm.h"
```

9. Введите обработчик сообщения для команды Файл – Новое окно. Для этого вызовите ClassWizard, выберите вкладку Message Maps, в поле Class name: установите CmainFrame, в поле Object IDs: выделите ID_FILE_TEXT, в поле Messages: – COMMAND. Нажмите кнопку Add Function… (рисунок 160).

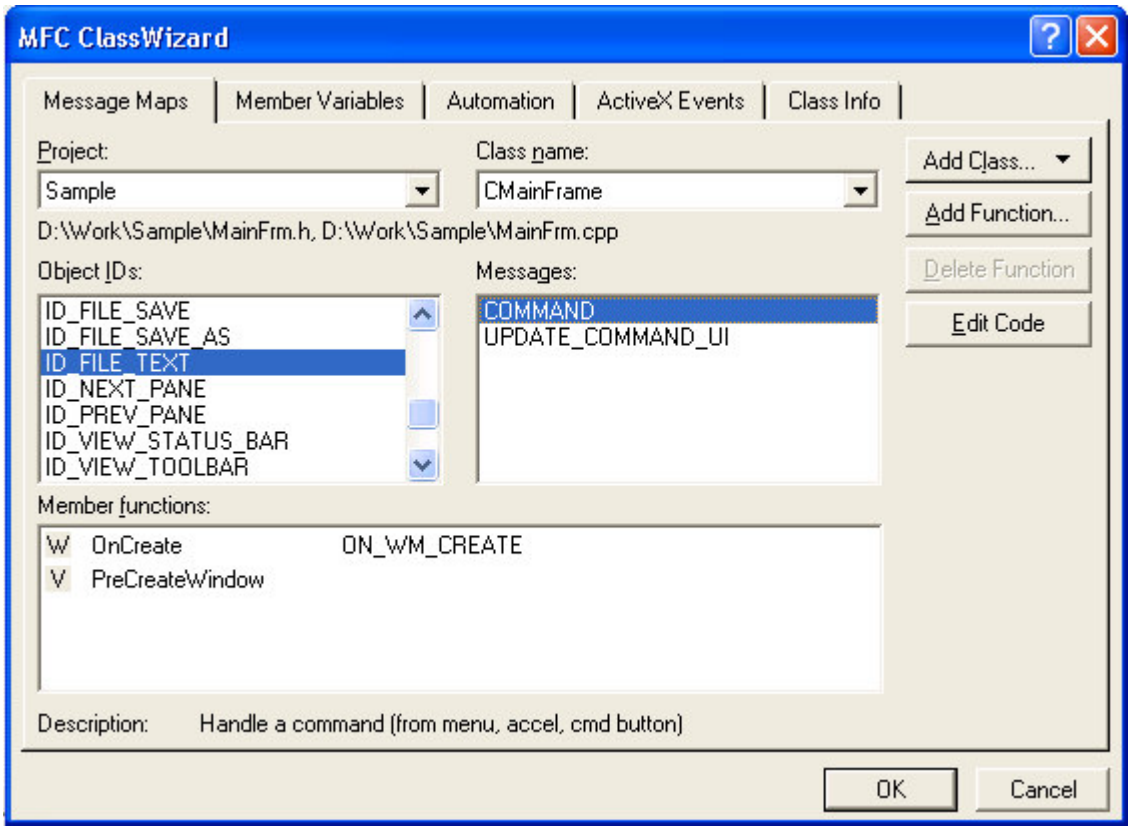



Рисунок 160. Обработчики сообщений приложения

Задайте имя обработчика сообщений OnFileText и нажмите кнопку OK (рисунок 161).

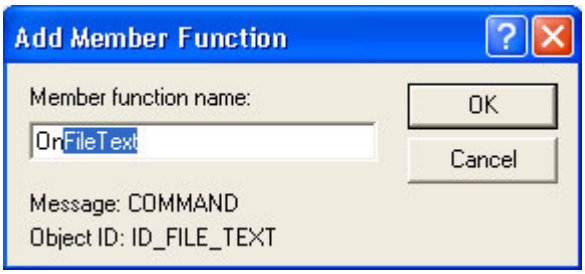



Рисунок 161. Задание имени обработчика сообщений

Нажмите кнопку Edit Code и введите код программы:

```
void CMainFrame::OnFileText()
{
    CChildFrame *pTextWnd = new CChildFrame;
    if (!pTextWnd->Create(lpszTextClass, _TEXT("Дочернее окно"),0, rectDefault, this))
    return;
}
```

10. Откройте описание класса дочернего окна CChildFrame. В разделе protected опишите переменную menu, используемую для подключения меню дочернего окна:

```
// Generated message map functions
```

```
protected:
    static CMenu menu;
```

В начале файла ChildFrm.cpp опишите переменную:

```
CMenu CChildFrame::menu;
```

Внесите изменения в процедуру BOOL CChildFrame::PreCreateWindow(CREATESTRUCT& cs):

```
BOOL CChildFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    cs.style |= WS_CHILD | WS_VISIBLE | WS_OVERLAPPEDWINDOW;
    if (menu.m_hMenu == NULL)
            menu.LoadMenu(IDR_SAMPLETYPE);
    m_hMenuShared = menu.m_hMenu;
    if( !CMDIChildWnd::PreCreateWindow(cs) )
            return FALSE;
    return TRUE;
}
```

11. Сохраните и проверьте работу приложения (рисунок 162).

Проверьте работу пунктов меню Окно (рисунок 163).

12. Введите обработчик команд IDM_BLACK – IDM_WHITE (один на всех) (рисунок 164).
13. Введите обработчик команд обновления IDM_BLACK – IDM_WHITE (один на всех) (рисунок 165).

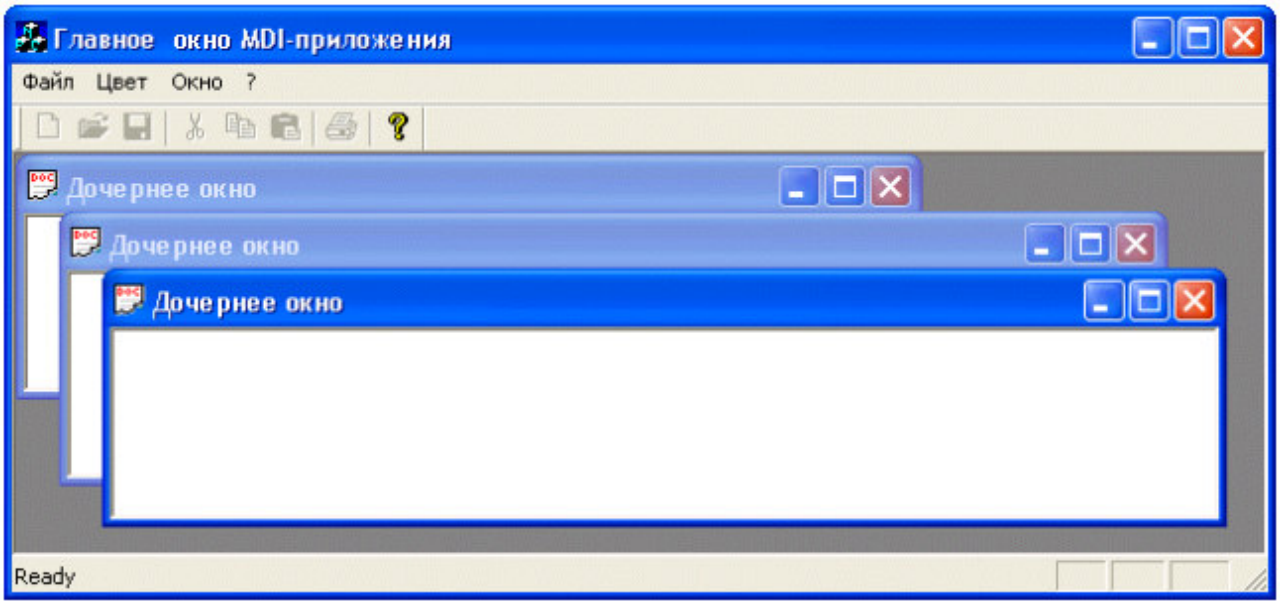



Рисунок 162. Проверка работы приложения

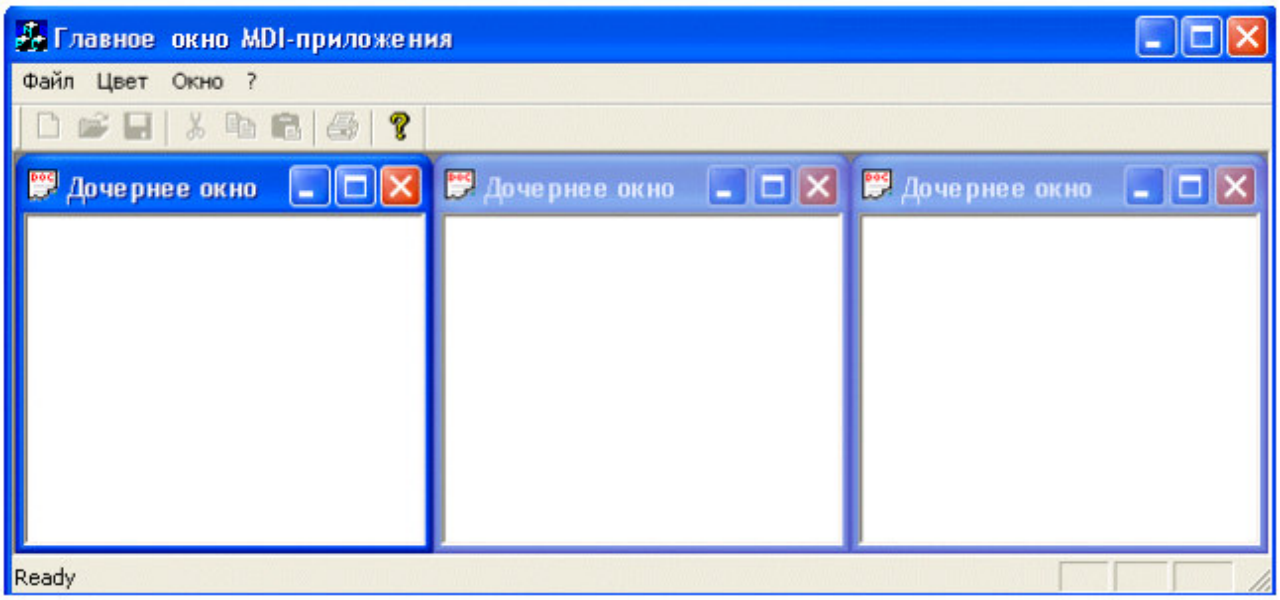



Рисунок 163. Проверка работы дочерних окон приложения

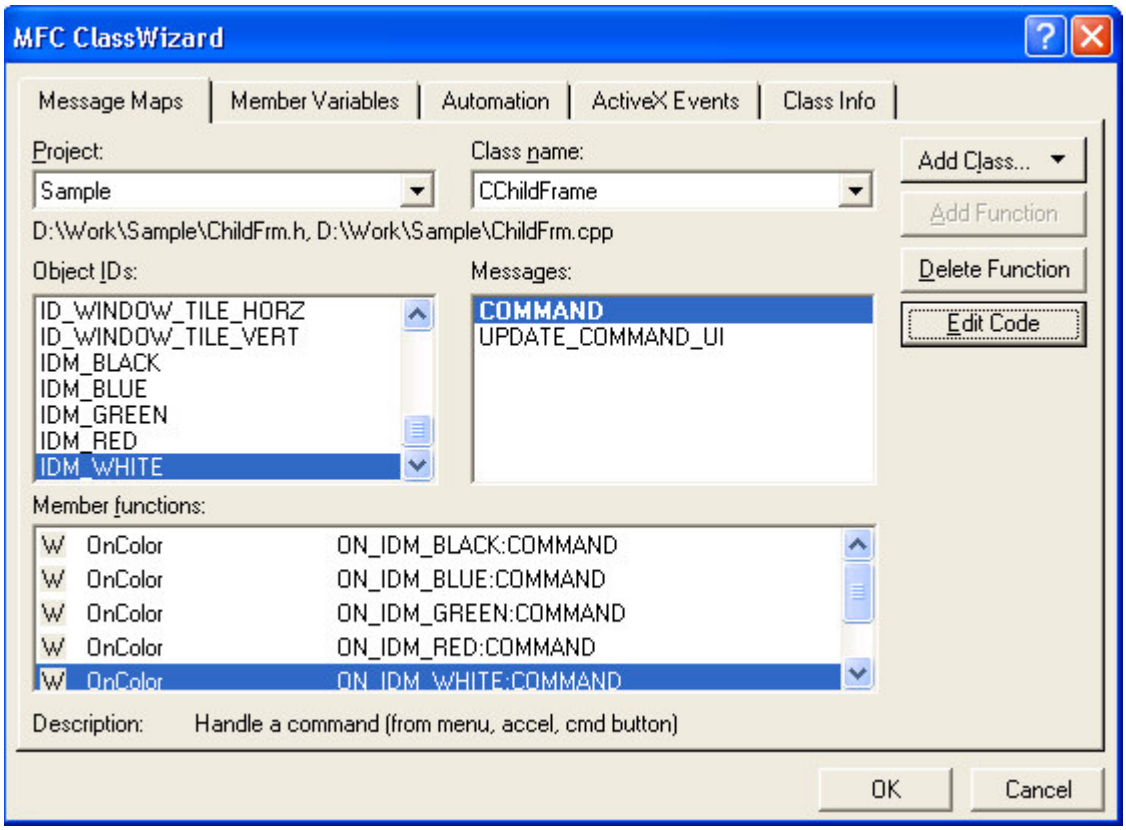



Рисунок 164. Установка обработчика команд IDM_BLACK – IDM_WHITE

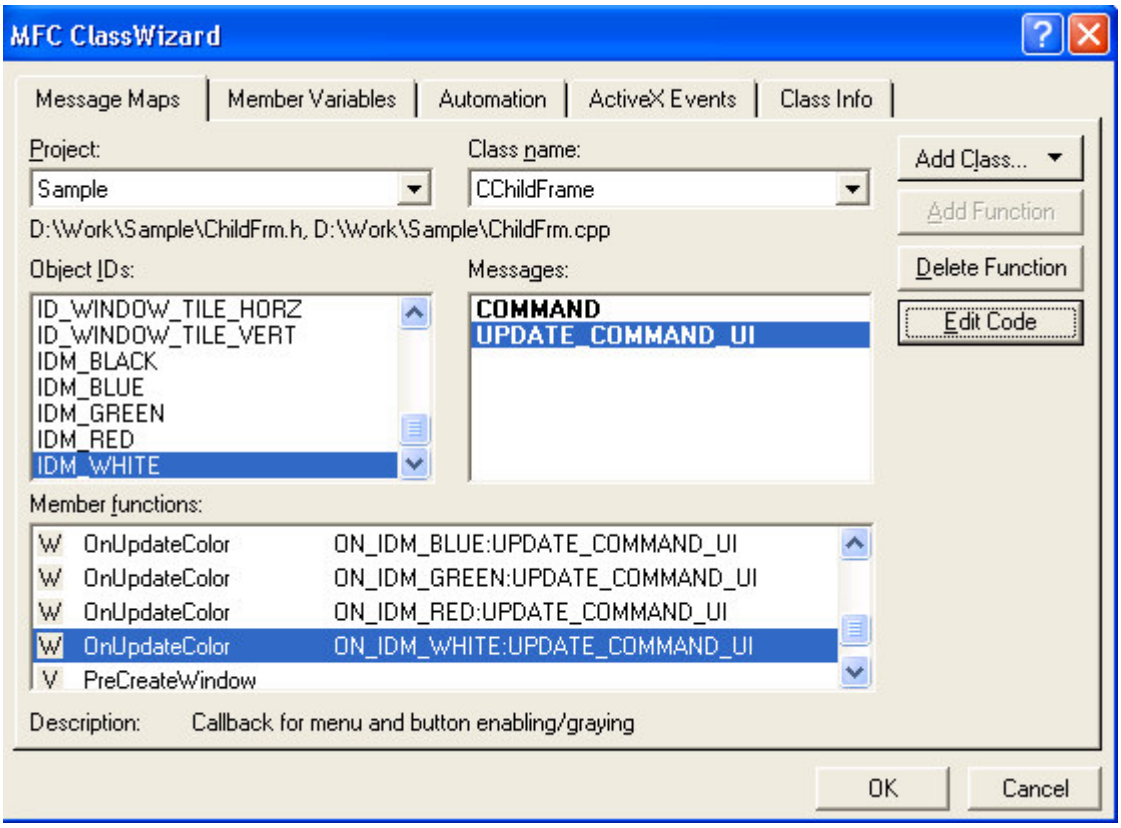



Рисунок 165. Установка обработчика команд обновления IDM_BLACK – IDM_WHITE

14. Откройте описание класса дочернего окна CChildFrame. В разделе protected опишите переменные:

```
// Generated message map functions
protected:
    static CMenu menu;
    UINT nIDColor;
    COLORREF clrText;
```

В начале файла ChildFrm.cpp опишите переменную:

```
COLORREF colorArray[] =
{
    RGB(0,0,0),
    RGB(255,255,255),
    RGB(255,0,0),
    RGB(0,0,255),
    RGB(0,255,0)
};
```

Отредактируйте созданные обработчики команд:

```
void CChildFrame::OnColor()
{
    // Получаем идентификатор выбранной команды
    nIDColor = LOWORD(GetCurrentMessage()->wParam);
    // По идентификатору команды выбираем из массива
    // соответствующее значение цвета
    clrText = colorArray[nIDColor - IDM_BLACK];
    // Перерисовать окно в ответ на сообщение WM_PAINT
    Invalidate();
}
```

```
void CChildFrame::OnUpdateColor(CCmdUI* pCmdUI)
{
    // Устанавливаем или убираем галочку у соответствующего элемента меню
    pCmdUI->SetCheck(pCmdUI->m_nID == nIDColor);
}
```

15. Добавьте обработчик сообщения WM_PAINT (рисунок 166).

Введите код обработчика сообщения WM_PAINT:

```
void CChildFrame::OnPaint()
{
    CPaintDC dc(this);          // Получаем контекст устройства
    CRect rect;
    GetClientRect(rect);
    // Устанавливаем текущие цвета текста и фона
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW));
    // Отображаем текст в центре окна
    dc.DrawText(_TEXT("Обработка сообщения WM_PAINT"),-1,rect,
            DT_SINGLELINE | DT_CENTER | DT_VCENTER);
}
```

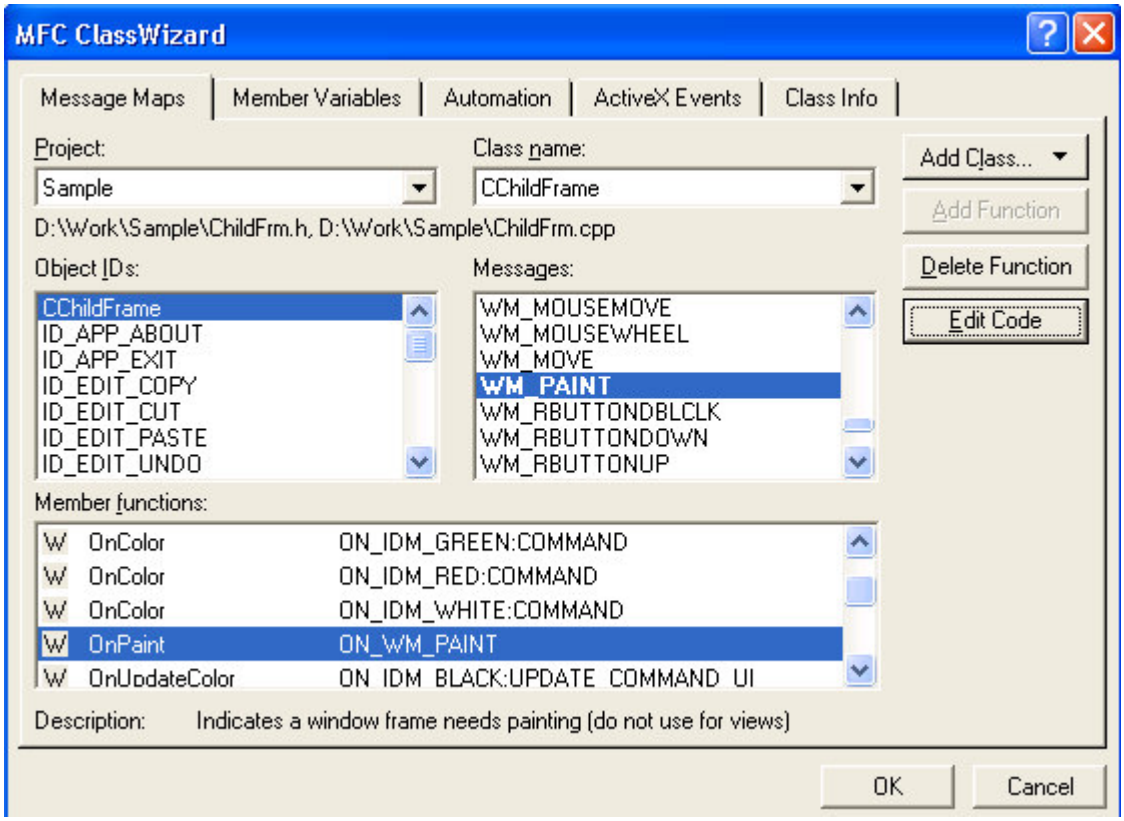



Рисунок 166. Добавление обработчика сообщения WM_PAINT

Внесите изменения в конструктор класса CChildFrame:

```
CChildFrame::CChildFrame()
{
nIDColor = IDM_BLACK;          // При запуске приложения устанавливаем
    clrText = RGB(0,0,0);      // черный цвет
}
```

16. Сохраните и проверьте работу приложения (рисунок 167).

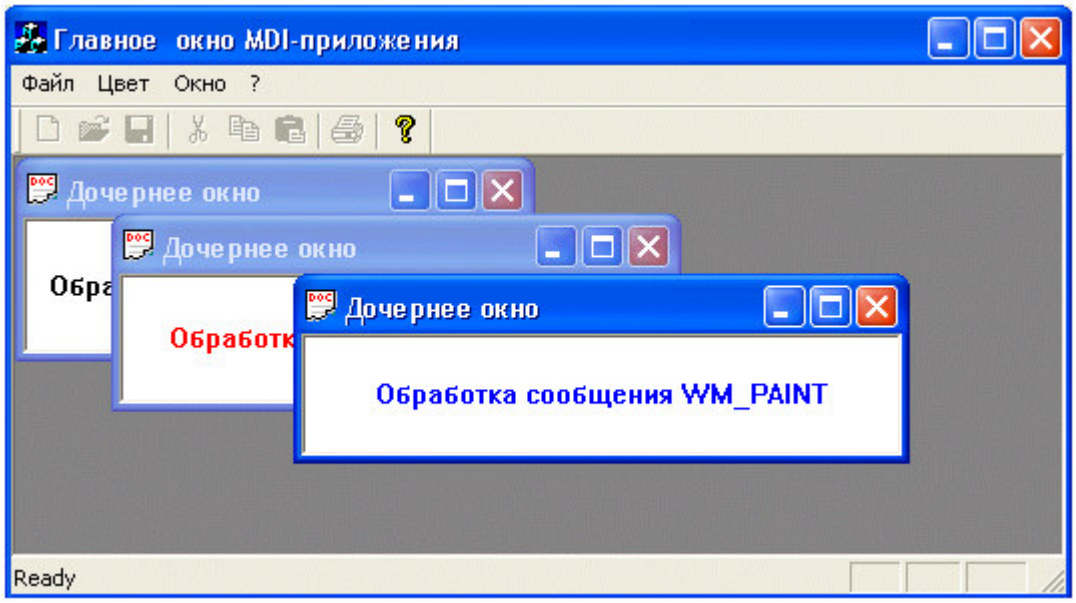


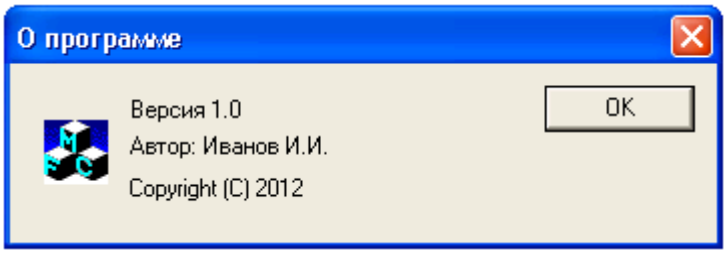

Рисунок 167. Проверка работы приложения

***Самостоятельная работа***

*Вариант 1*

Внесите в созданное приложение следующие изменения.

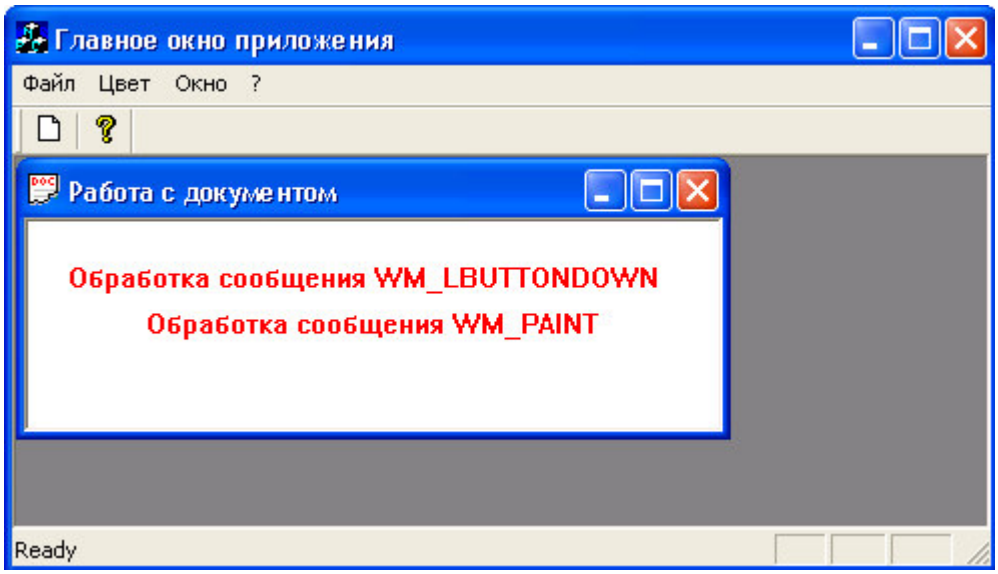
1. Измените диалоговое окно IDD_ABOUTBOX в соответствии с рисунком:




2. Измените и сделайте функциональным панель инструментов приложения в соответствии с рисунком:

3. Измените заголовки главного и дочернего окна в соответствии с рисунком, приведенным ниже.

4. Добавьте к дочернему окну приложения обработчик нажатия левой кнопки мыши, выводящей на экран сообщение «Обработка сообщения WM_LBUTTONDOWN»:




Текст соответствующего кода программы:

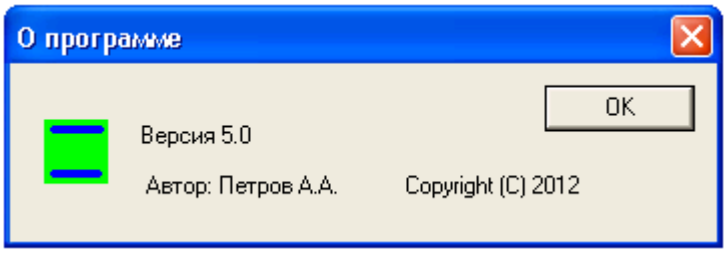
```
void CChildFrame::OnLButtonDown(UINT nFlags, CPoint point)
{
    TCHAR strText[] = " Обработка сообщения WM_LBUTTONDOWN";
    CClientDC dc(this);
    CRect rect;
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW));
    dc.TextOut(20,20,strText, strlen(strText));
    CMDIChildWnd::OnLButtonDown(nFlags, point);
}
```

5. Проанализируйте отличие обработчиков сообщений WM_PAINT и WM_LBUTTONDOWN.

*Вариант 2*

Внесите в созданное приложение следующие изменения.

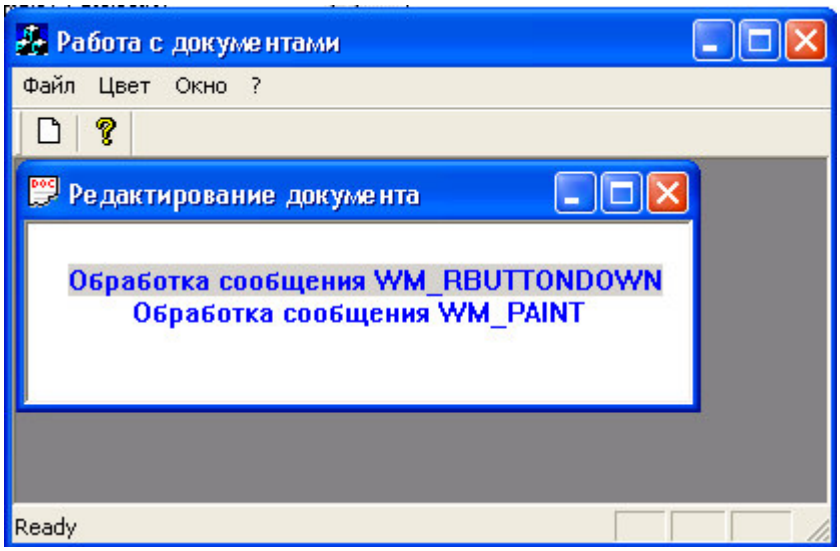
1. Измените диалоговое окно IDD_ABOUTBOX в соответствии с рисунком:




2. Измените и сделайте функциональным панель инструментов приложения в соответствии с рисунком:

3. Измените заголовки главного и дочернего окна в соответствии с рисунком, приведенным ниже.

4. Добавьте к дочернему окну приложения обработчик нажатия правой кнопки мыши, выводящей на экран сообщение «Обработка сообщения WM_RBUTTONDOWN»:




Текст соответствующего кода программы:

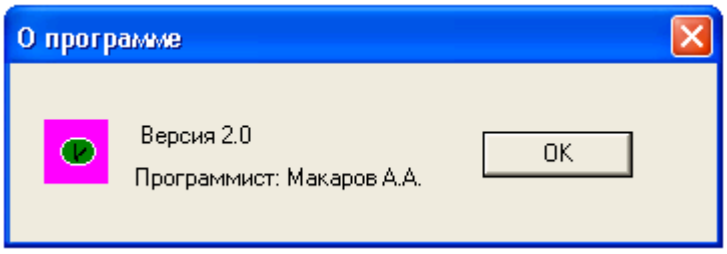
```
void CChildFrame::OnRButtonDown(UINT nFlags, CPoint point)
{
    TCHAR strText[] = "Обработка сообщения WM_RBUTTONDOWN";
    CClientDC dc(this);
    CRect rect;
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW+5));
    dc.TextOut(20,20,strText, strlen(strText));
    CMDIChildWnd::OnRButtonDown(nFlags, point);
}
```

5. Проанализируйте отличие обработчиков сообщений WM_PAINT и WM_RBUTTONDOWN.

*Вариант 3*

Внесите в созданное приложение следующие изменения.

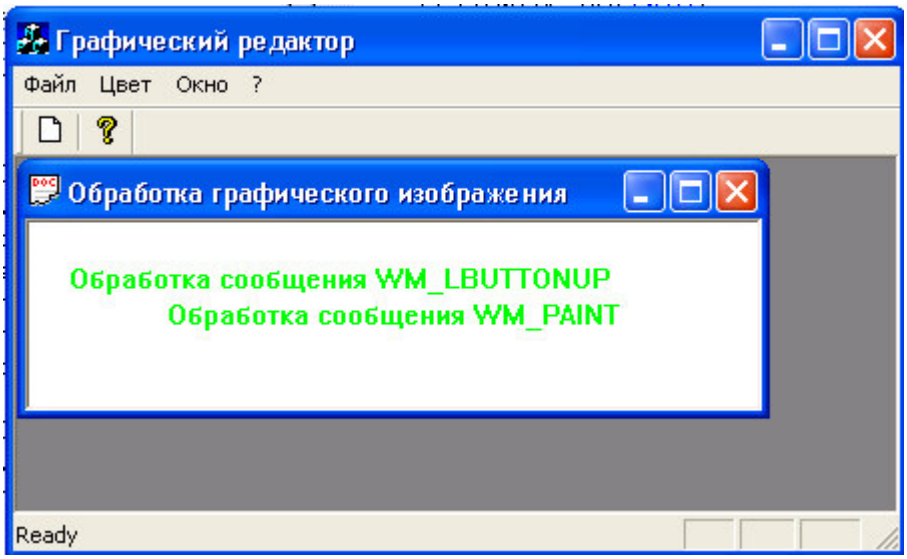
1. Измените диалоговое окно IDD_ABOUTBOX в соответствии с рисунком:




2. Измените и сделайте функциональным панель инструментов приложения в соответствии с рисунком:

3. Измените заголовки главного и дочернего окна в соответствии с рисунком, приведенным ниже.

4. Добавьте к дочернему окну приложения обработчик нажатия левой кнопки мыши, выводящей на экран сообщение «Обработка сообщения WM_LBUTTONUP»:




Текст соответствующего кода программы:

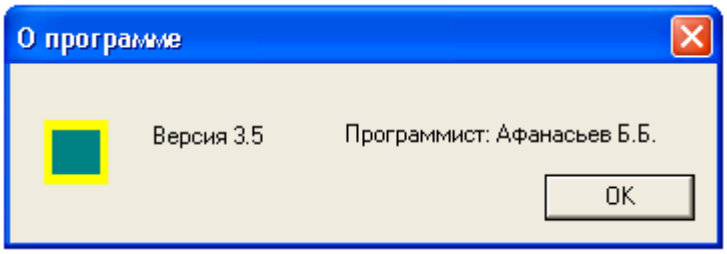
```
void CChildFrame::OnLButtonUp(UINT nFlags, CPoint point)
{
    TCHAR strText[] = "Обработка сообщения WM_LBUTTONUP";
    CClientDC dc(this);
    CRect rect;
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW));
    dc.TextOut(20,20,strText, strlen(strText));
    CMDIChildWnd::OnLButtonUp(nFlags, point);
}
```

5. Проанализируйте отличие обработчиков сообщений WM_PAINT и WM_LBUTTONUP.

*Вариант 4*

Внесите в созданное приложение следующие изменения.

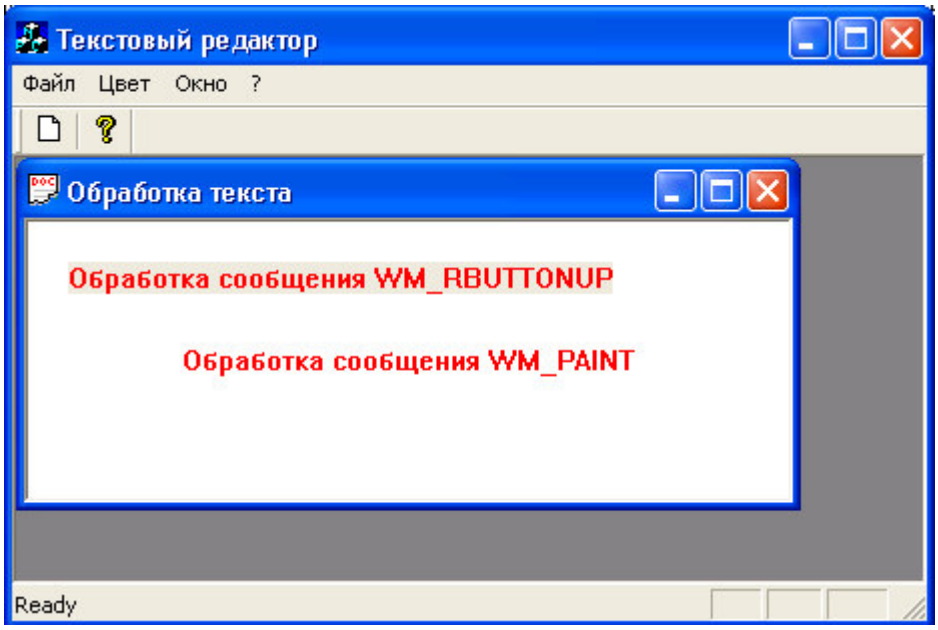
1. Измените диалоговое окно IDD_ABOUTBOX в соответствии с рисунком:




2. Измените и сделайте функциональным панель инструментов приложения в соответствии с рисунком:

3. Измените заголовки главного и дочернего окна в соответствии с рисунком, приведенным ниже.

4. Добавьте к дочернему окну приложения обработчик нажатия правой кнопки мыши, выводящей на экран сообщение «Обработка сообщения WM_RBUTTONUP»:




Текст соответствующего кода программы:

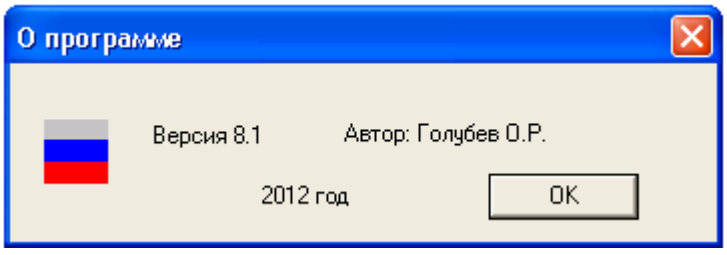
```
void CChildFrame::OnRButtonUp(UINT nFlags, CPoint point)
{
    TCHAR strText[] = "Обработка сообщения WM_RBUTTONUP";
    CClientDC dc(this);
    CRect rect;
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW+10));
    dc.TextOut(20,20,strText, strlen(strText));
    CMDIChildWnd::OnLButtonUp(nFlags, point);
}
```

5. Проанализируйте отличие обработчиков сообщений WM_PAINT и WM_RBUTTONUP.

*Вариант 5*

Внесите в созданное приложение следующие изменения.

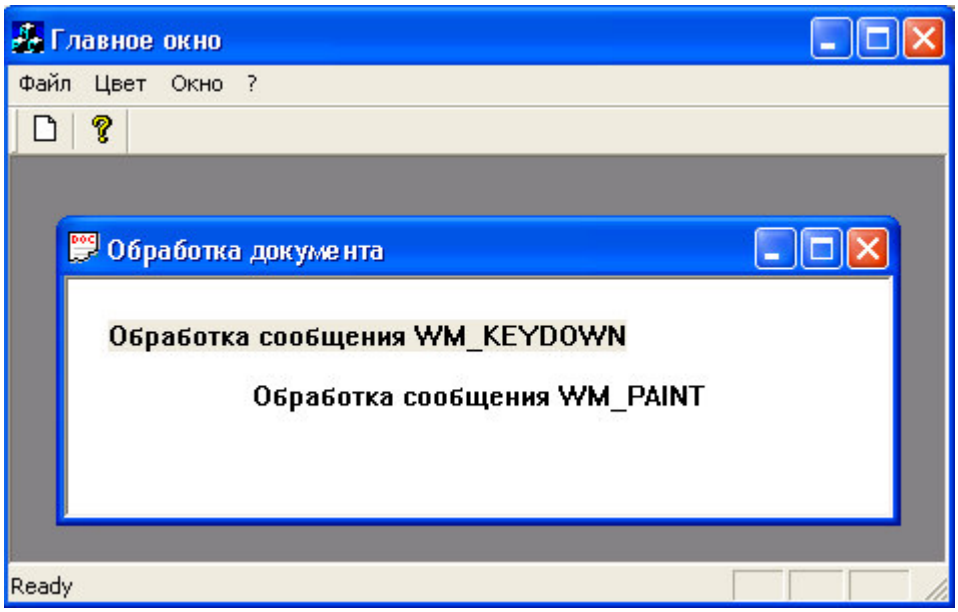
1. Измените диалоговое окно IDD_ABOUTBOX в соответствии с рисунком:




2. Измените и сделайте функциональным панель инструментов приложения в соответствии с рисунком:

3. Измените заголовки главного и дочернего окна в соответствии с рисунком, приведенным ниже.

4. Добавьте к дочернему окну приложения обработчик нажатия любой клавиши клавиатуры, выводящей на экран сообщение «Обработка сообщения WM_KEYDOWN»:




Текст соответствующего кода программы:

```
void CChildFrame::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
    TCHAR strText[] = "Обработка сообщения WM_KEYDOWN";
    CClientDC dc(this);
    CRect rect;
    dc.SetTextColor(clrText);
    dc.SetBkColor(::GetSysColor(COLOR_WINDOW+10));
    dc.TextOut(20,20,strText, strlen(strText));
    CMDIChildWnd::OnKeyDown(nChar, nRepCnt, nFlags);
}
```

5. Проанализируйте отличие обработчиков сообщений WM_PAINT и WM_KEYDOWN.

**Лабораторный практикум № 19. Использование и разработка библиотек динамической компоновки в языке C++**

***Продолжительность: 90 минут.***

***Дисциплина «Программирование». Юнита 9.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель лабораторного практикума:*** изучение основных принципов работы с библиотеками динамической компоновки в языке C++.

***Вводная часть***

*Использование библиотек динамической компоновки*

В каждом приложении Windows используются библиотеки динамической компоновки - DLL (Dynamic Link Library). В DLL содержаться все функции Win32 API и множество других функций операционных систем Win32.

DLL – это просто наборы функций, классов или ресурсов, собранные в библиотеки. Но в отличие от статических библиотек (файлов .lib), библиотеки DLL не присоединены непосредственно к выполняемым файлам. В программе есть информация об их местонахождении, и эти библиотеки загружаются целиком в момент выполнения программы. Разные процессы могут пользоваться совместно одними и теми же

библиотеками, находящимися в памяти. Такой подход позволяет сократить объём памяти, необходимый для нескольких приложений, использующих много общих библиотек, а также контролировать размеры exe-файлов.

При запуске приложение пытается найти все файлы DLL, неявно подключённые к приложению, и поместить их в оперативную память процесса. При любом методе загрузки поиск файлов DLL осуществляется в следующей последовательности:

• каталог, в котором находится exe-файл;

• текущий каталог процесса;

• системные каталоги Windows;

• каталоги, определенные в переменной окружения PATH.

Если библиотека DLL не найдена, приложение выводит сообщение об ошибке. Если библиотека найдена, она помещается в оперативную память процесса и остается там до его окончания, а приложение может обращаться к функциям, содержащимся в DLL.

Неявная загрузка и использование модуля DLL осуществляется так же просто, как и работа с обычными функциями. При компиляции модуля DLL транслятор Microsoft Visual Studio создаёт наряду с файлом .dll ещё файл с расширением .lib. Файл библиотеки .lib используется для разрешения адресов загрузки в модуле DLL и содержит полное имя динамически подключаемой библиотеки, тогда как файл заголовка содержит описание модуля библиотеки. Для неявной загрузки библиотеки файл заголовка надо подключить к исходному файлу, использующему функции или классы из DLL, и указать имя файла библиотеки (.lib) в поле Object/Library modules на закладке Link диалогового окна Project Settings, доступ к которому можно получить, выбрав команду Project – Settings. Если файл .lib не находится в каталоге проекта, то надо указать полный путь к нему.

*Динамическая загрузка DLL*

Динамическая загрузка DLL позволяет программе определить, какая из библиотек и когда будет загружаться, а также определить действия при неудачной загрузке библиотеки. В начале надо поместить модуль библиотеки в память процесса. Данная операция выполняется с помощью функции LoadLibrary(), имеющий единственный аргумент - имя загружаемого модуля:

```
HMODULE hDll;
hDll = LoadLibrary(“MyLib”);
if (hDll = = NULL)
        // обработка ошибки загрузки DLL
```

Стандартным расширением библиотеки считается .dll, если не указывается другое. В приведённом примере Windows будет искать файл MyLib.dll. Если в имени файла указан путь, то только он будет использоваться при поиске библиотеки. Если такая библиотека уже загружена в память, то будет просто возвращён её дескриптор. Зная дескриптор библиотеки, можно загружать из неё ресурсы.

Для определения адресов отдельных функций внутри библиотеки используется функция

```
FARPROC
GetProcAddress(HMODULE hModule, LPCSTR lpProcName);
```

где hModule - дескриптор библиотеки DLL,

lpProcName - имя функции.

Пример:

```
HMODULE hDll;
UINT (*pfnFunc) (char* str);
UINT res;
hDll = LoadLibrary(“MyLib”);
pfnFunc = (UINT(*) (char*)):: GetProcAddress(hDll, “MyFunc”);
if (pfnFunc != NULL)
   res = (*pfnFunc) (“Test string”);
```

Можно также сослаться на функцию по порядковому номеру, по которому она экспортируется:

```
pfnFunc = (UINT(*) (char*))::GetProcAddress(hDll, MAKEINTRESOURCE(12));
```

Если функция не обнаружена, GetProcAddress() возвращает NULL, иначе возвращается указатель на функцию. В приведённом примере ищется функция, которая принимает указатель на char и возвращает значение типа UINT.

Для выгрузки библиотеки из памяти процесса используется функция FreeLibrary(), которой в качестве параметра передаётся дескриптор библиотеки.

При работе с библиотеками DLL в рамках MFC, для загрузки и выгрузки библиотек надо использовать функции AfxLoadLibrary() и AfxFreeLibrary().

*Создание DLL*

Проще всего создать новый проект DLL с помощью мастера AppWizard. Для простых библиотек можно выбрать тип проекта Win32 Dynamic-Link Library. Новому проекту будут присвоены все необходимые параметры для создания библиотеки DLL. Файлы исходных текстов придётся добавлять в проект вручную.

Если планируется использовать MFC, то лучше выбрать тип проекта MFC Application (dll). В проект будут добавлены необходимые ссылки на библиотеки MFC и файлы исходных текстов, содержащий описание и реализацию в библиотеке DLL объекта класса приложения, производного от CWinApp.

Большинство библиотек DLL - просто набор функций, экспортируемых в приложения. Кроме функций, предназначенных для экспортирования, в каждой библиотеке есть функция DllMain(). Эта функция предназначена для инициализации и очистки библиотеки. Структура функции DllMain(), создаваемая AppWizard для проекта MFC Application (dll), выглядит так:

```
// Глобальная структура, в которой хранится текущее состояние DLL
static AFX_EXTENSION_MODULE MyDllDLL = { NULL, NULL };
extern "C" int APIENTRY
DllMain(HINSTANCE hInstance, DWORD dwReason, LPVOID lpReserved)
{
    UNREFERENCED_PARAMETER(lpReserved);
    if (dwReason == DLL_PROCESS_ATTACH)
    {
            TRACE0("MYDLL.DLL Initializing!\n");
            // Однократная инициализация DLL-расширения
            if (!AfxInitExtensionModule(MyDllDLL, hInstance))
                    return 0;
            new CDynLinkLibrary(MyDllDLL);
    }
    else if (dwReason == DLL_PROCESS_DETACH)
    {
            TRACE0("MYDLL.DLL Terminating!\n");
            // Завершает выполнение до вызова деструкторов
            AfxTermExtensionModule(MyDllDLL);
    }
    return 1;
}
```

Первый параметр DllMain, поданный системой, представляет собой Windows-описатель DLL. Его можно использовать при вызове функций, требующих этот описатель, например GetModuleFileName. Второй параметр указывает причину вызова DLL. Он может принимать одно из следующих четырех значений:

DLL_PROCESS_ATTACH – указывает на то, что DLL загружается в виртуальное адресное пространство процесса, так как стартовал сам процесс или была вызвана функция LoadLibrary. Второй случай – это явный вызов DLL;

DLL_THREAD_ATTACH – указывает на то, что текущий процесс создает новый поток (thread). В этот момент система вызывает все DLL, которые уже загружены в пространстве процесса;

DLL_THREAD_DETACH – указывает на то, что поток завершается и DLL может освободить динамические ресурсы, связанные с данным потоком (если они были);

DLL_PROCESS_DETACH – указывает на то, что DLL выгружается из адресного пространства процесса либо в результате завершения процесса, либо потому, что процесс вызвал функцию FreeLibrary. В этом случае DLL может освободить память.

Параметр lpReserved зарезервирован для внутреннего использования Windows. При динамической загрузке DLL он будет равен NULL.

Если DllMain вернет FALSE или 0, то клиентское приложение завершится с кодом ошибки.

Файл .def содержит имя и описание библиотеки, а также список экспортируемых функций:

```
LIBRARY      "MyDll"
DESCRIPTION  'MyDll Windows Dynamic Link Library'
EXPORTS
   ; Экспортируемые функции
    sum @10
```

В строке экспорта функции можно указать её порядковый номер, поставив перед ним символ @. Этот номер будет затем использоваться при вызове функции GetProcAddress(). По умолчанию, компилятор присваивает порядковые номера всем экспортируемым объектам, но эти номера присваиваются непредсказуемо. Параметр NONAME запрещает компилятору включать имя функции в таблицу экспортирования DLL.

Для экспортирования классов можно в объявлении класса воспользоваться макромодификатором AFX_CLASS_EXPORT.

***Практическая часть***

*Задача.* Разработать приложение, вычисляющее сумму двух целых чисел. Для вычисления суммы использовать библиотеку динамической компоновки.

Этап 1. Разработать основу приложения, предназначенного для вычисления суммы двух чисел.

1. Запустите Microsoft Visual Studio.
2. Выполните команду File – New… Выберите вкладку Projects.
3. Выберите тип проекта MFC AppWizard(exe). В поле Location установите рабочую директорию. В поле Project name введите имя проекта (рисунок 168).
4. Нажмите на кнопку OK. После нажатия кнопки OK инструмент AppWizard будет показывать страницы диалога для уточнения проекта (шаги определения установок проекта). Сделайте выбор в соответствии с таблицей 21.
5.

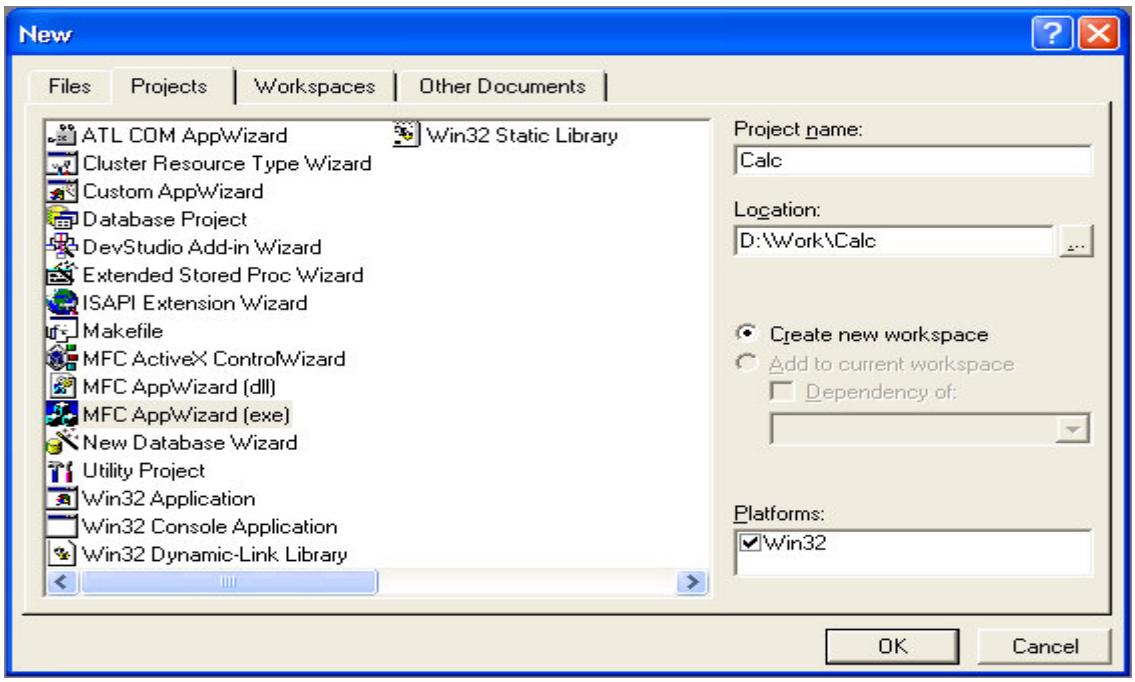
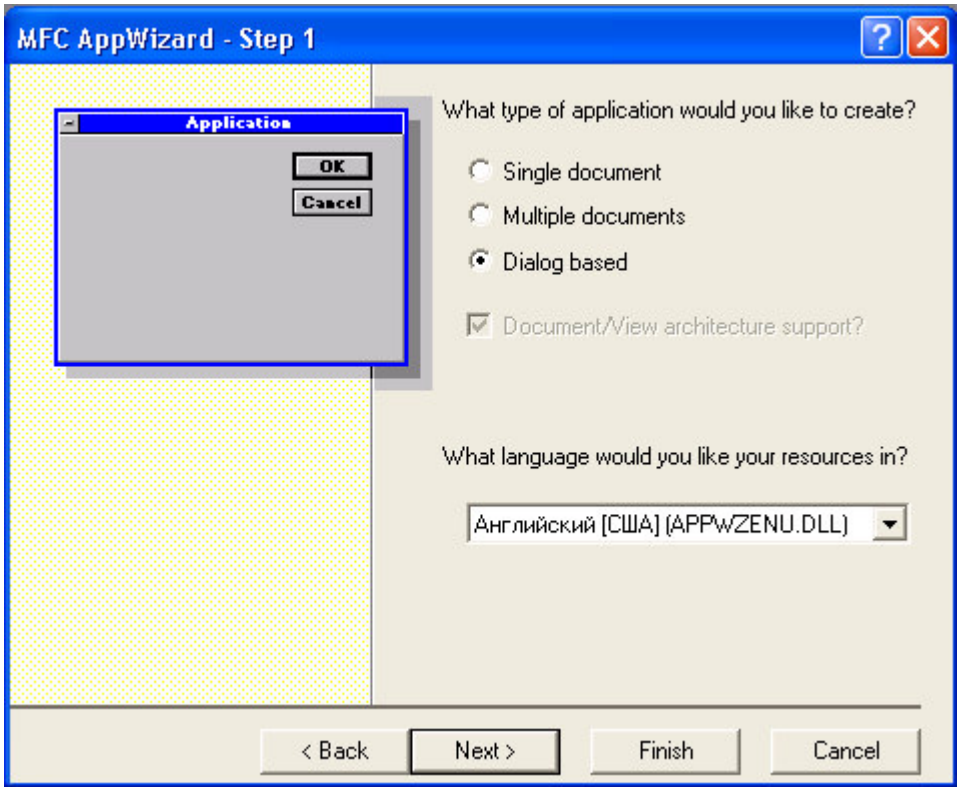
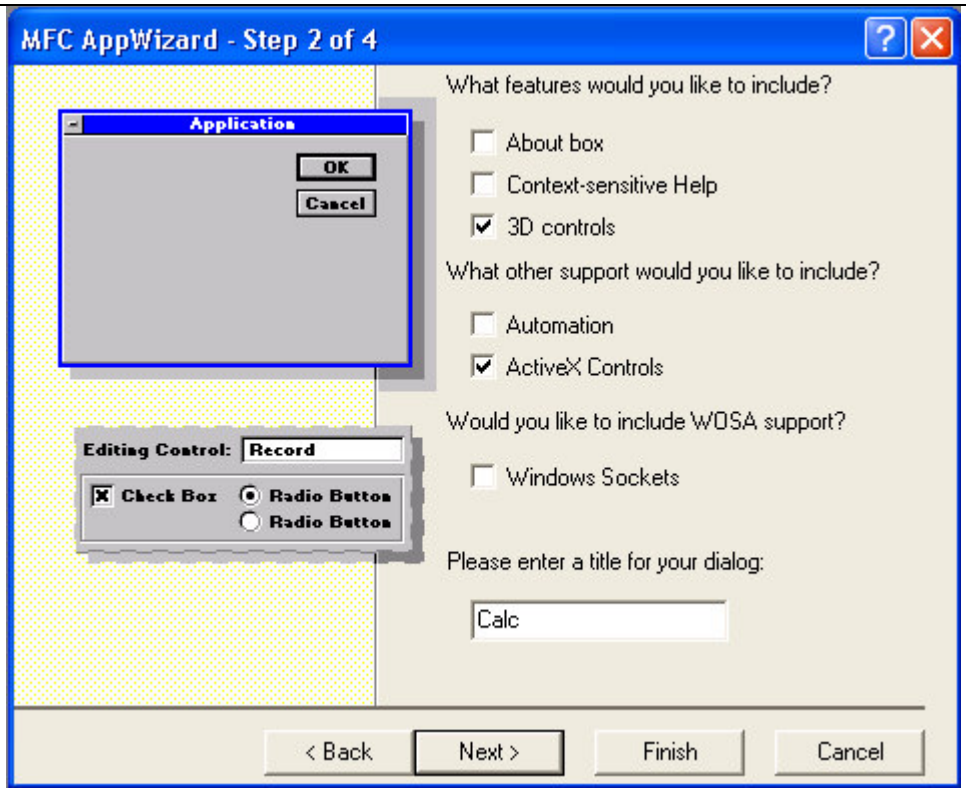
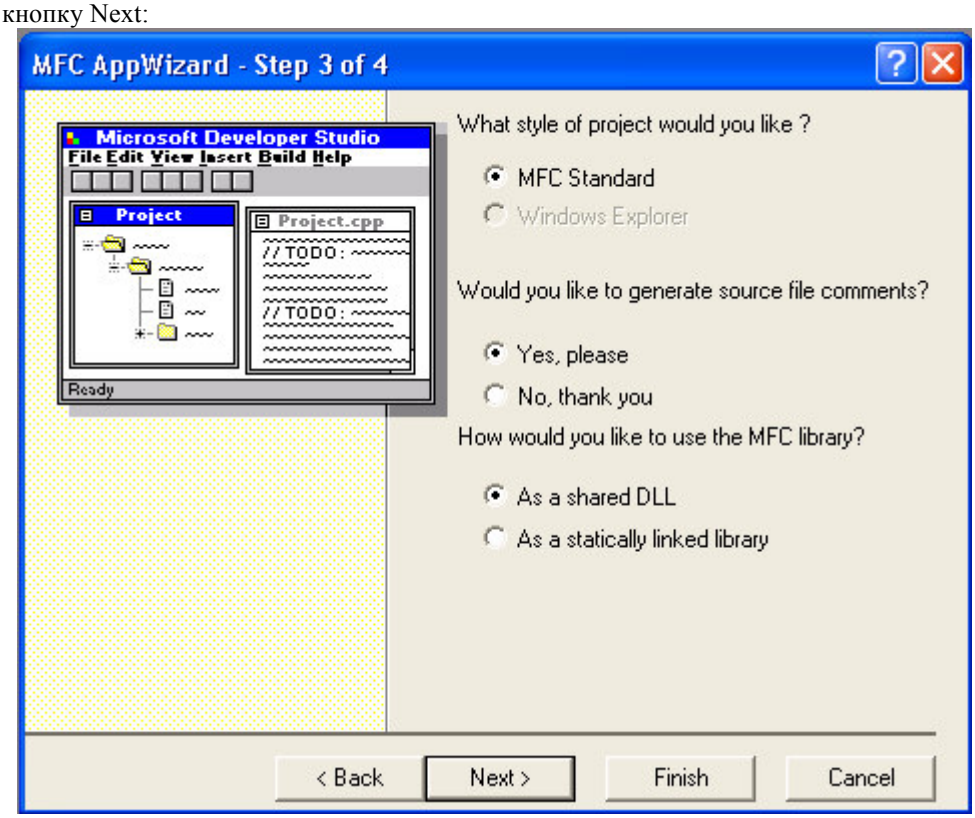
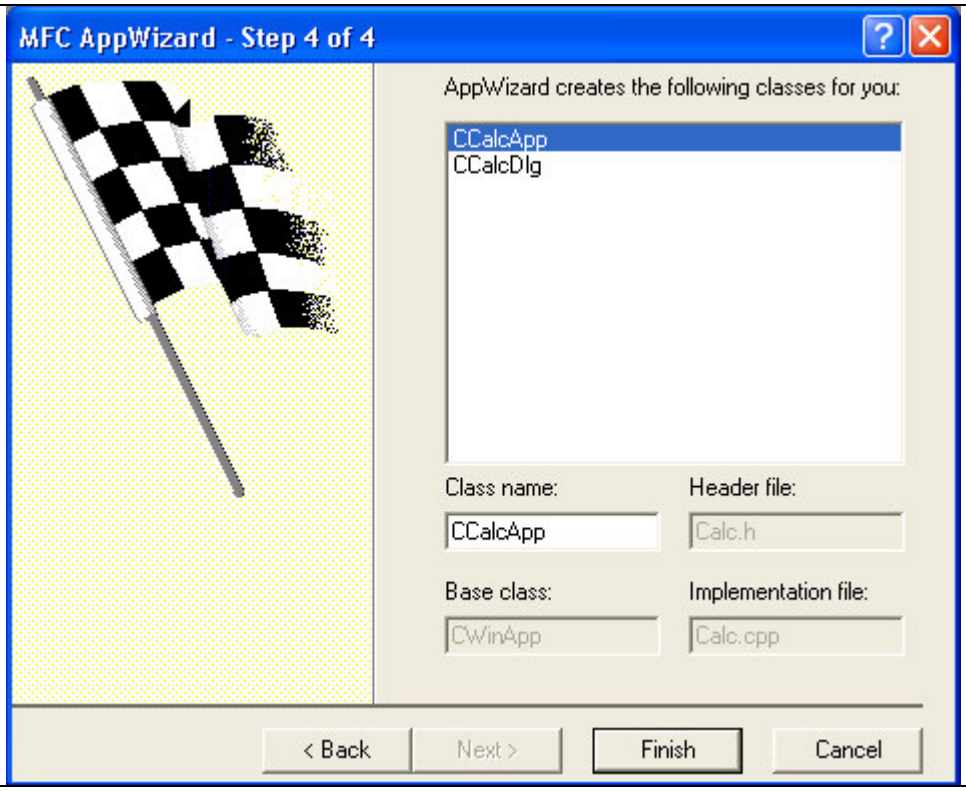



Рисунок 168. Выбор тип приложения

Таблица 21. Этапы генерации проекта

| Шаг | Выбор или действие |
|---|---|
| 1 (Выбор типа приложения) | Выберите переключатель Dialog based (на базе окна диалога). Нажмите на кнопку Next:   |
| 2 (Настройка | Выключите About box. Нажмите на кнопку Next: |

| Шаг | Выбор или действие |
|---|---|
| возможностей создаваемого приложения) |  |
| 3 (Подключение библиотеки MFC) | Выберите переключатели MFC Standard; Yes, please; As a shared Dll. Нажмите на кнопку Next:<br>  |
| 4 (Создание классов приложения) | Нажмите на кнопку Finish: |

| Шаг | Выбор или действие |
|---|---|
| |  |



6. Visual Studio отобразит диалоговое окно New Project Information (информация о новом проекте) (рисунок 169).

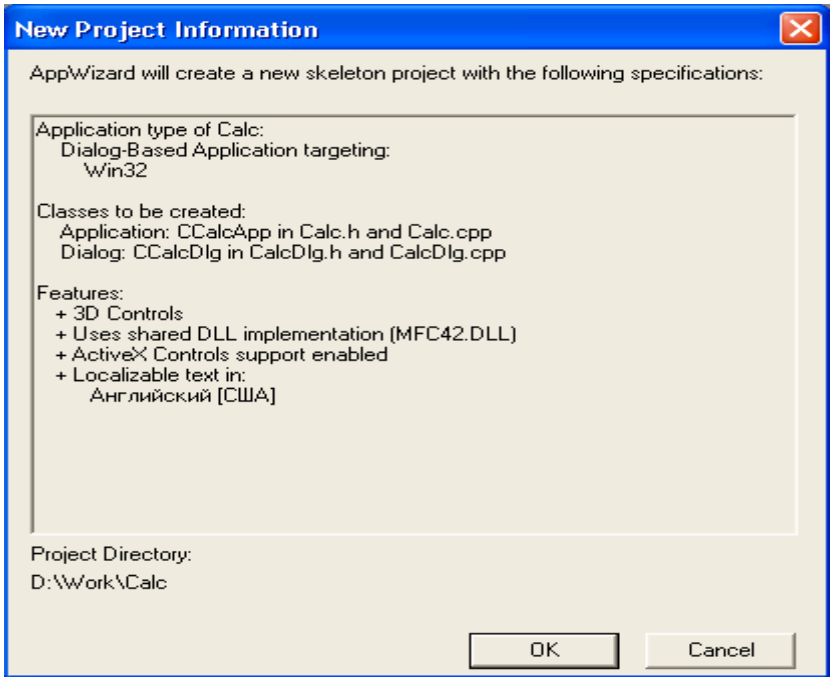




Рисунок 169. Завершение генерации приложения

Убедитесь, что все установки вашей программы корректны (при необходимости нажмите на кнопу Cancel и внесите изменения). Нажмите на кнопку OK. MFC AppWizard создаст оболочку программы.

7. Перейдите в окно ResourceView. Раскройте элемент дерева Dialog. Выделите элемент IDD_CALC_DIALOG. Для установки русского языка, находясь на этом элементе, в контекстном меню выберите Properties – Language : Russian (рисунок 170).

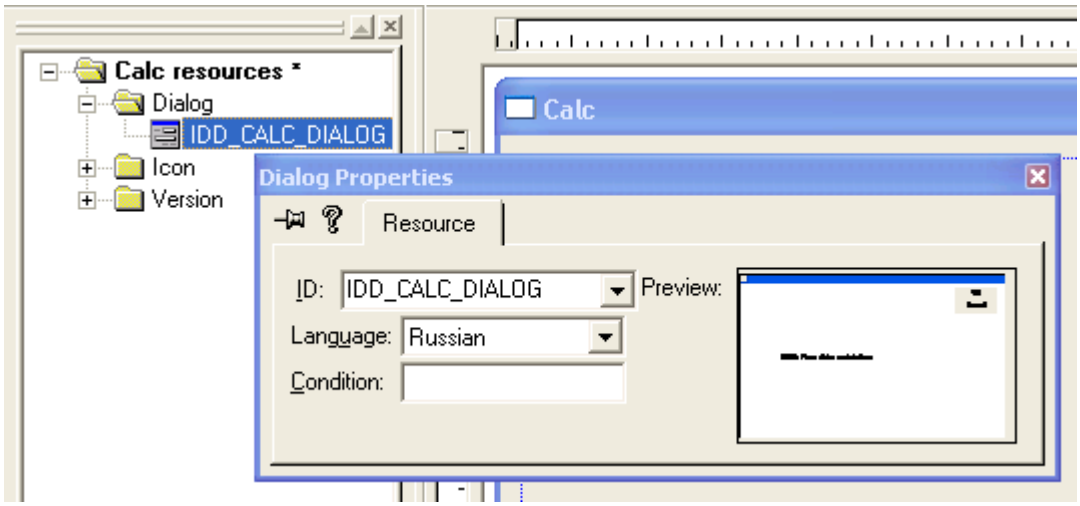



Рисунок 170. Установка русского языка

8. Сохраните приложение.

9. С помощью панели инструментов Contols вставьте в диалоговое окно элементы управления в соответствии с рисунком 171.

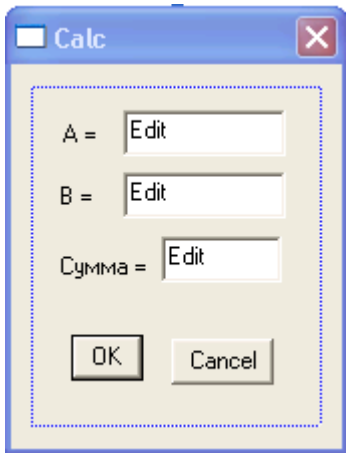



Рисунок 171. Формирование формы приложения

10. Введите в класс CCalcDlg переменные, позволяющие управлять содержимым окон. Для этого:

– выполните команду View – ClassWizard…;
– выберите опцию Select an existing class; нажмите на кнопку OK;
– укажите имя класса CCalcDlg, нажмите на кнопку Select;
– выберите вкладку Member Variables;
– выберите идентификатор IDC_EDIT1 и нажмите на кнопку Add Variable;
– заполните поля диалога в соответствии с таблицей:

| Поле | Идентификатор переменной |
|---|---|
| Member Variable name: | m_a |
| Category: | Value |
| Variable type: | int |

– повторите эти действия для поля редактирования IDC_EDIT2, задав имя переменной m_b, и поля редактирования IDC_EDIT3, задав имя переменной m_sum;

Список переменных будет следующим (рисунок 172).

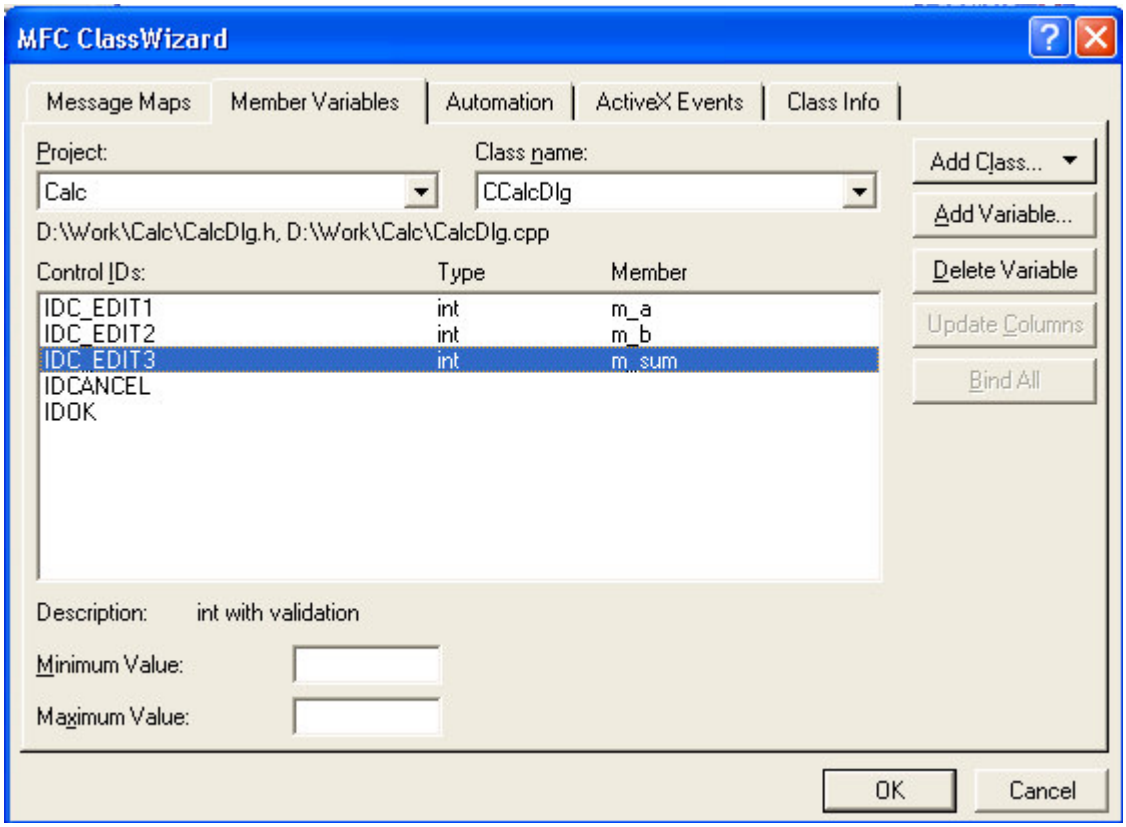



Рисунок 172. Список переменных приложения

11. Сохраните и запустите приложение (рисунок 173).

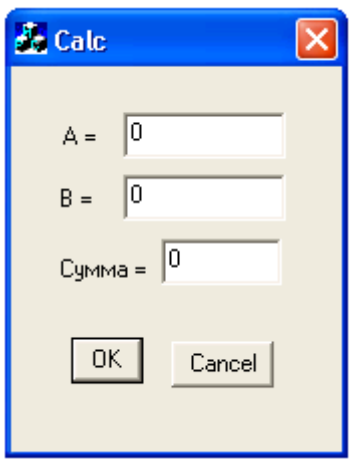


Рисунок 173. Проверка работы приложения

12. Закройте проект.

Этап 2. Разработать DLL-библиотеку, содержащую экспортируемую функцию, предназначенную для вычисления суммы двух чисел.

1. Выполните команду File – New… Выберите вкладку Projects.

2. Выберите тип проекта MFC AppWizard(dll). В поле Location установите рабочую директорию. В поле Project name введите имя DLL-библиотеки (рисунок 174).

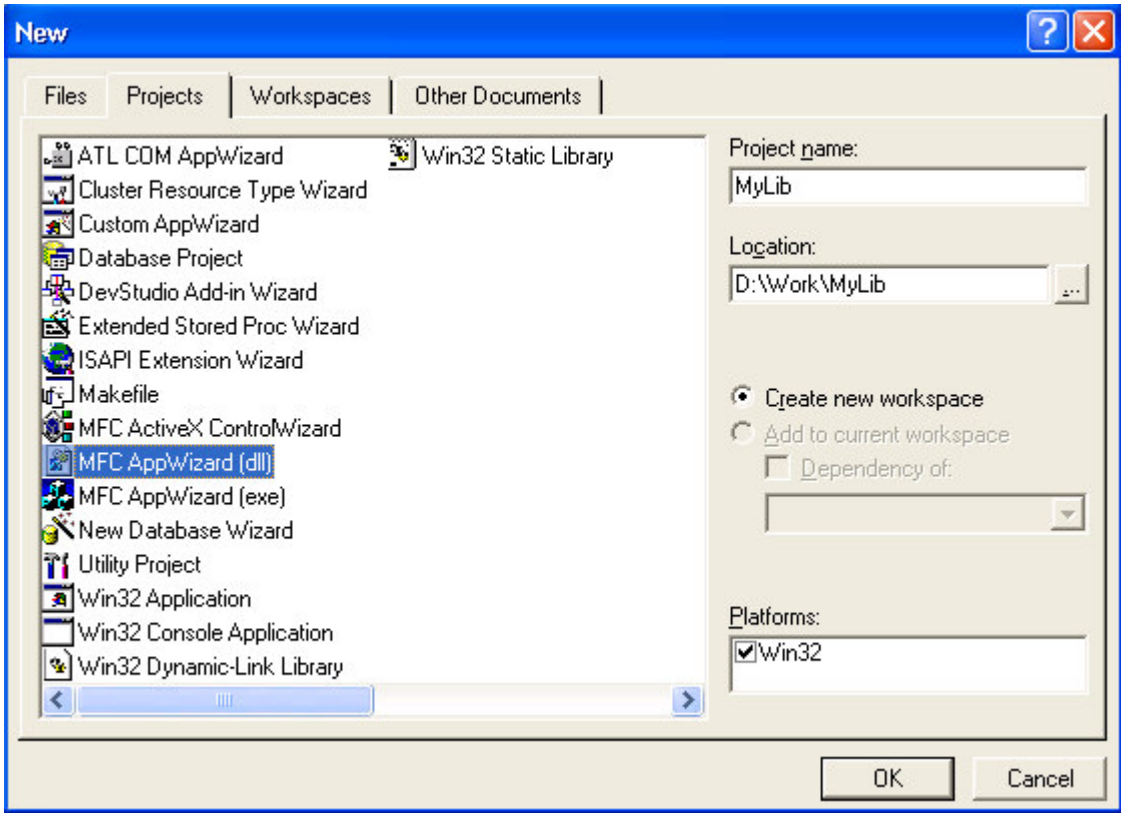
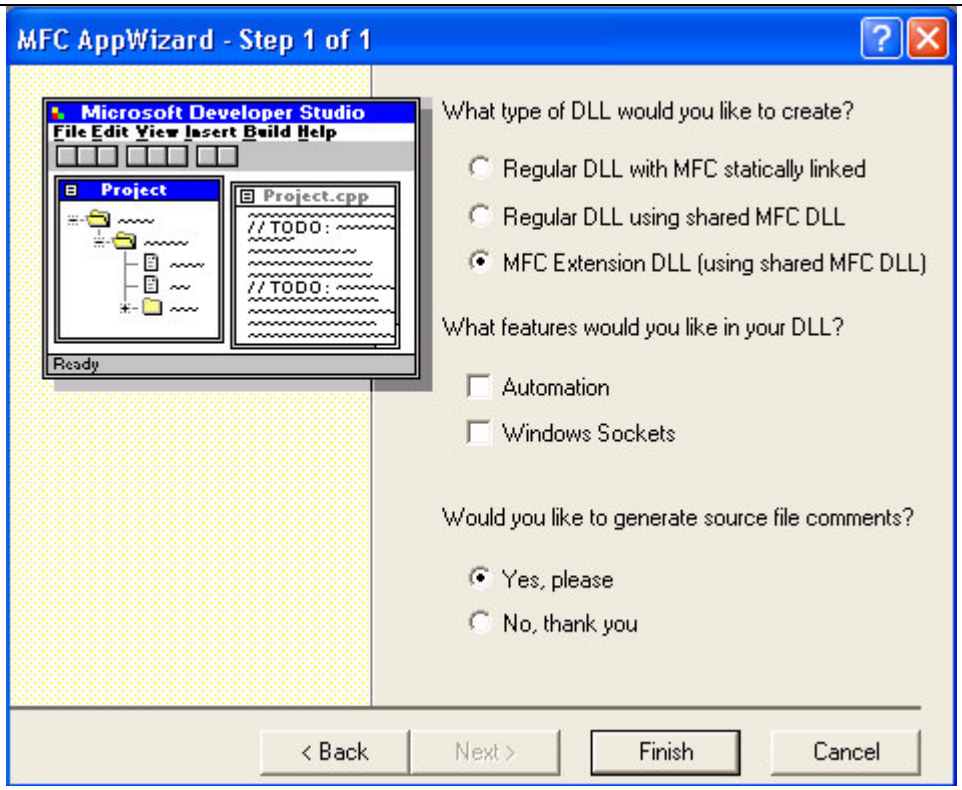



Рисунок 174. Выбор типа приложения

3. Нажмите на кнопку OK. После нажатия кнопки OK инструмент AppWizard будет показывать страницы диалога для уточнения проекта (шаги определения установок проекта). Сделайте выбор в соответствии с таблицей 22.

Таблица 22. Этапы разработки DLL-библиотеки

| Шаг | Выбор или действие |
|---|---|
| 1 (Выбор типа DLL-библиотеки) | Выберите переключатель MFC Extension DLL (using shared MFC DLL). Нажмите на кнопку Finish: |

| Шаг | Выбор или действие |
|---|---|
| |  |



4. Visual Studio отобразит диалоговое окно New Project Information (информация о новом проекте) (рисунок 175).

Убедитесь, что все установки вашей программы корректны (при необходимости нажмите на кнопу Cancel и внесите изменения). Нажмите на кнопку OK. MFC AppWizard создаст оболочку DLL-библиотеки.

Созданные файлы приложения можно посмотреть в окне FileView (рисунок 176).

Ознакомьтесь с содержимым файлов MyLib.cpp и MyLib.def.

5. Сохраните и запустите проект. В окне Executable For Debug Session введите имя созданного на этапе 1 исполняемого файла (рисунок 177).

6. Нажмите на кнопку OK. Закройте запущенное приложение.

7. Откройте файл MyLib.cpp. Добавьте в конце модуля экспортируемую функцию, вычисляющую сумму двух целых чисел:

```
int sum(int a, int b) {return a+b;}
```

8. Откройте файл MyLib.def. Добавьте функцию, вычисляющую сумму двух целых чисел, в список экспортируемых функций:

```
; MyLib.def : Declares the module parameters for the DLL.
LIBRARY      "MyLib"
DESCRIPTION  'MyLib Windows Dynamic Link Library'
```

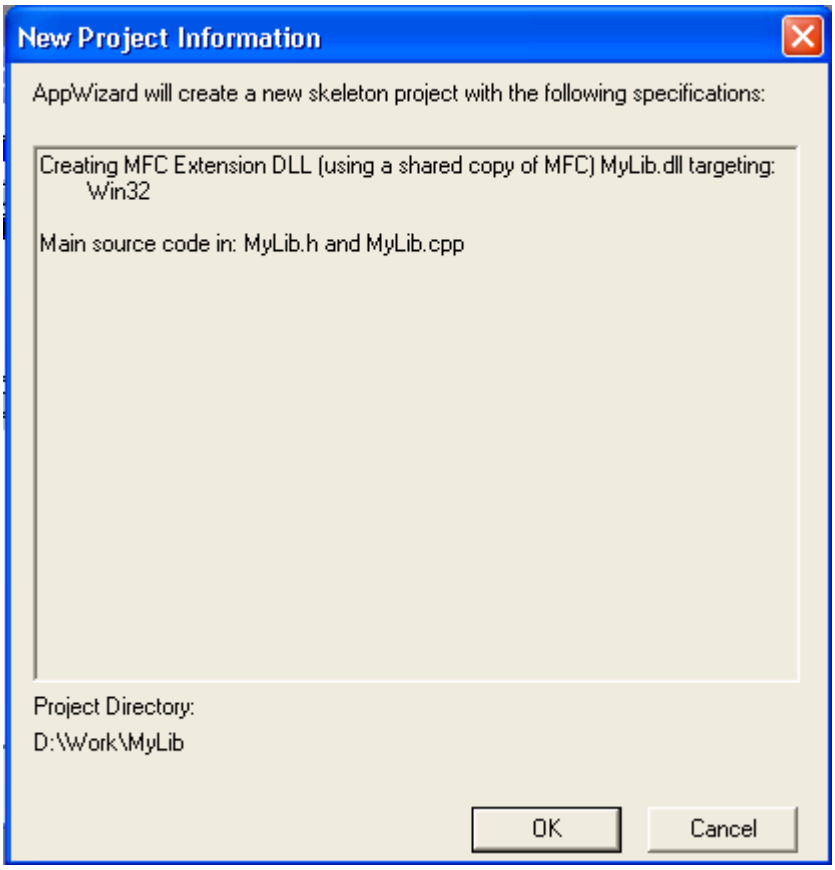



Рисунок 175. Завершение генерации DLL-библиотеки

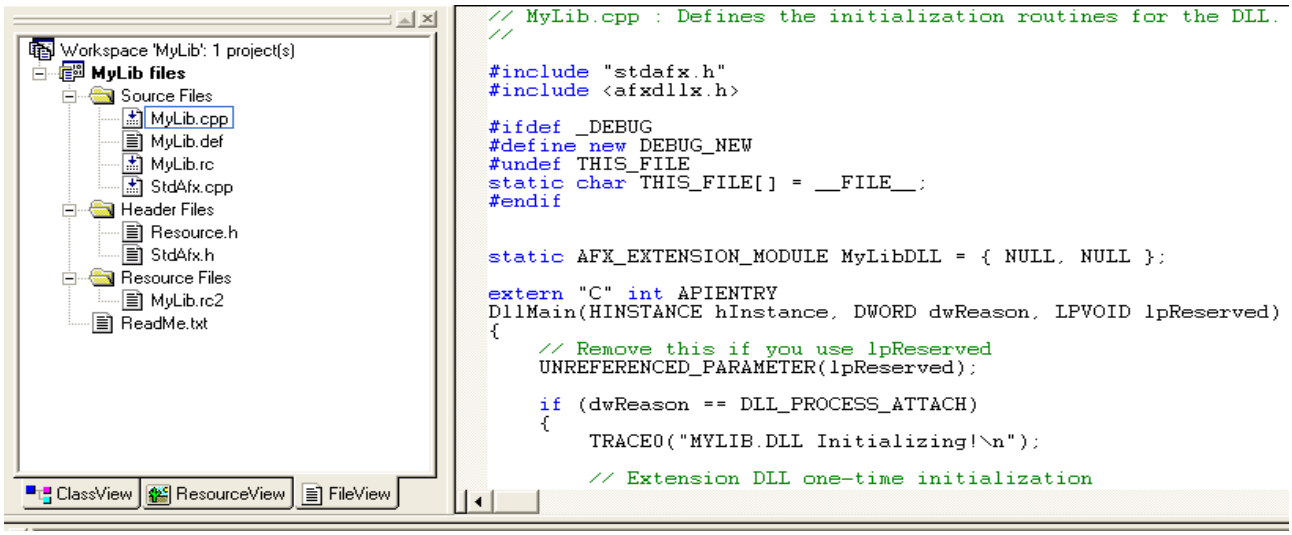



Рисунок 176. Перечень файлов DLL-библиотеки

```
EXPORTS
   ; Explicit exports can go here
    sum @10
```

9. Сохраните и перекомпилируйте DLL-библиотеку.
10. Закройте проект.

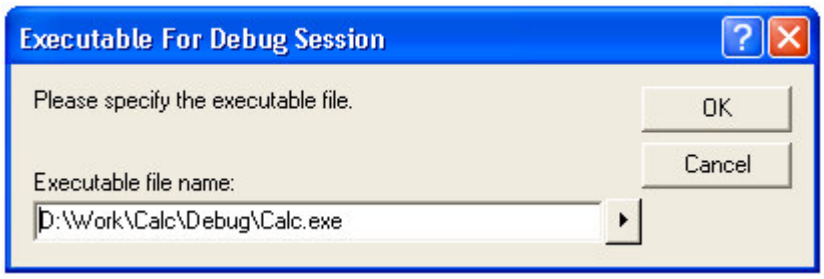



Рисунок 177. Задание имени созданного на этапе 1 исполняемого файла

Этап 3. Разработка основной части приложения, вызывающего DLL-библиотеку.

1. Откройте проект Calc.
2. Добавьте обработчик сообщений для кнопки OK:

```
void CCalcDlg::OnOK()
{
    UpdateData(TRUE);
    int nSum;
    HMODULE hLib = LoadLibrary("D:\\Work\\MyLib\\Debug\\MyLib.dll");
    ASSERT(hLib != NULL);
    int (*psum) (int a, int b);
    psum = (int (*) (int,int)) GetProcAddress(hLib,"sum");
    if (psum != NULL) nSum = (*psum) (m_a,m_b);
    m_sum = nSum;
    UpdateData(FALSE);
    FreeLibrary(hLib);
}
```

3. Сохраните и запустите приложение. Проверьте работу DLL-библиотеки (рисунок 178).

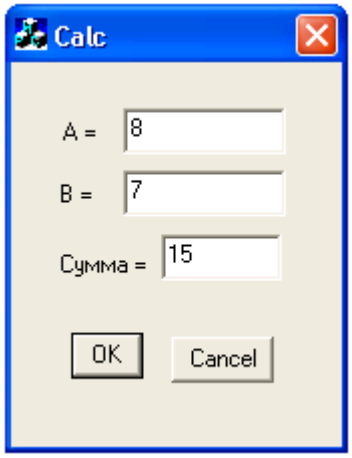



Рисунок 178. Проверка работы DLL-библиотеки

***Самостоятельная работа***

*Вариант 1*

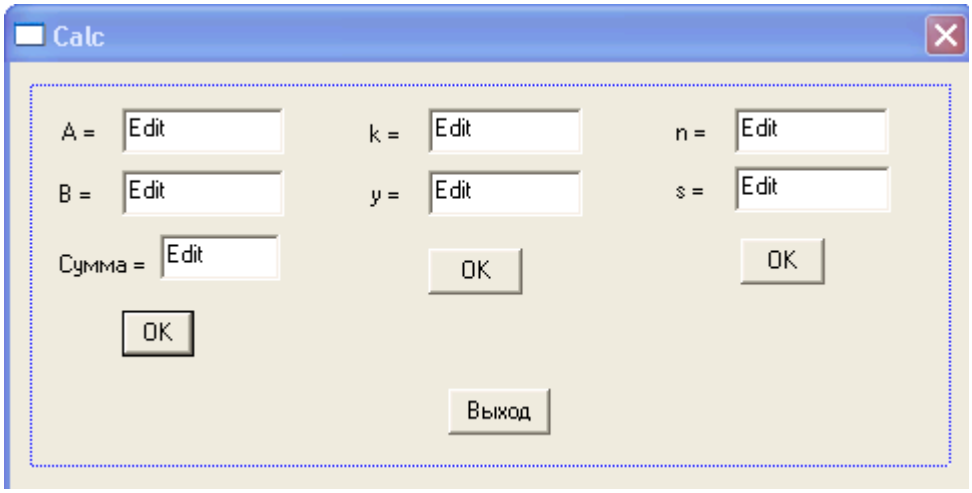
1. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double y(double k), вычисляющую значение *y* для заданного k:

$$y = \begin{Bmatrix} -k^2, если\ k \ge 7 \\ \dfrac{-2k}{k^2-9}, если\ k < 7 \end{Bmatrix}.$$

2. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double s(int n), вычисляющую значение s для заданного n:

$$s = \prod_{k=2}^{n} \frac{k}{k+1}.$$

3. Добавьте в проект Calc обработчики сообщений, вызывающие из библиотеки MyLib.dll функции double y(double k) и double s(int n):




*Вариант 2*

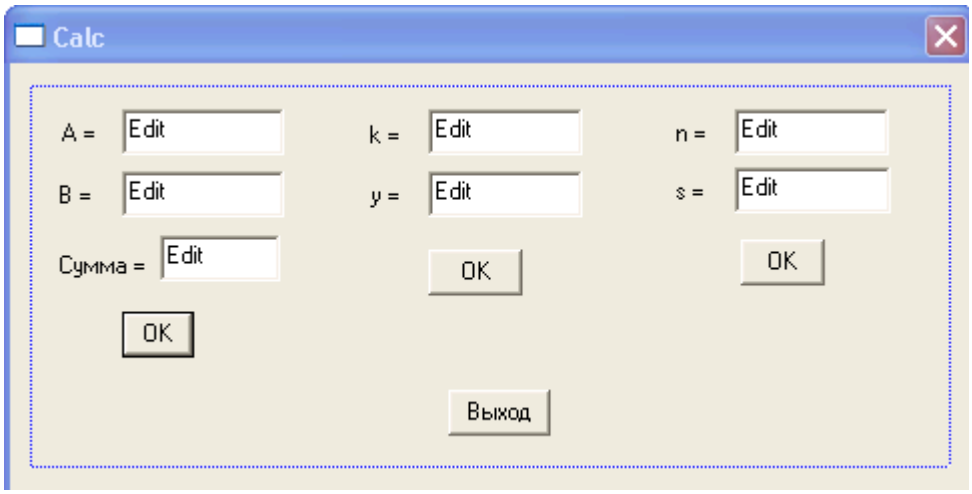
1. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double y(double k), вычисляющую значение *y* для заданного k:

$$y = \begin{Bmatrix} 9-k, \text{если } k > 1 \\ \dfrac{\sin 3k}{k+1}, \text{если } k \le 1 \end{Bmatrix}.$$

2. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double s(int n), вычисляющую значение s для заданного n:

$$\mathrm{s} = \prod_{\mathrm{k}=1}^{\mathrm{n}} \frac{\mathrm{k}+1}{\mathrm{k}+2}.$$

3. Добавьте в проект Calc обработчики сообщений, вызывающие из библиотеки MyLib.dll функции double y(double k) и double s(int n):




*Вариант 3*

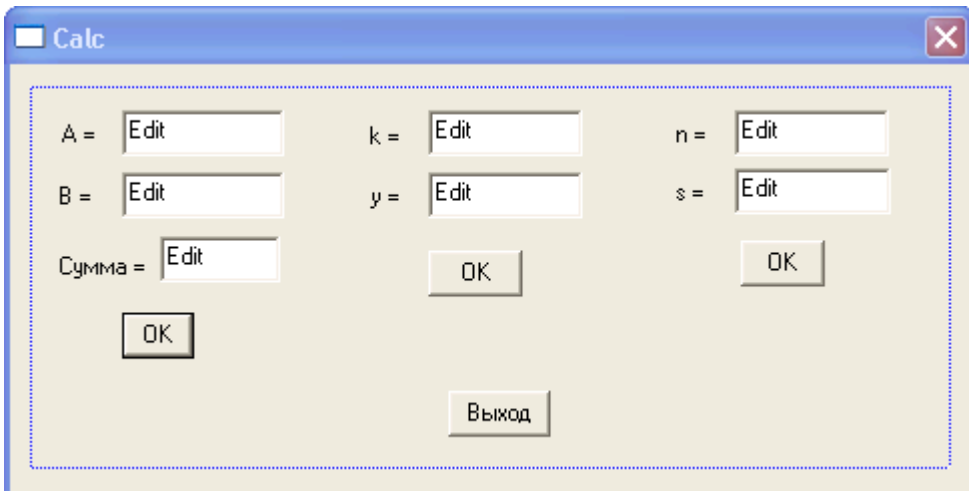
1. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double y(double k), вычисляющую значение *y* для заданного k:

$$y = \begin{cases} -k^2 + k - 9, \text{если } k \geq 8 \\ \dfrac{1}{k^3 - 6}, \text{если } k < 8 \end{cases}.$$

2. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double s(int n), вычисляющую значение s для заданного n:

$$s = \sum_{k=1}^{n} \frac{1}{2k+1}.$$

3. Добавьте в проект Calc обработчики сообщений, вызывающие из библиотеки MyLib.dll функции double y(double k) и double s(int n):




*Вариант 4*

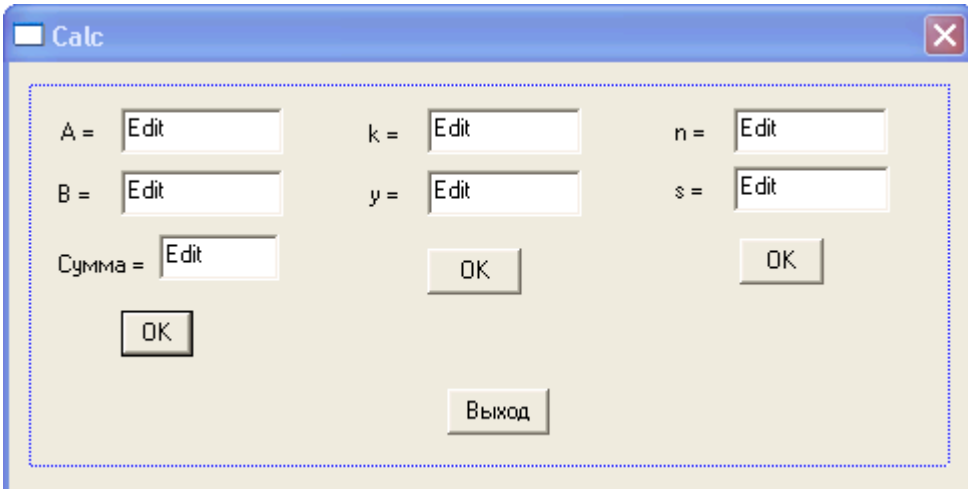
1. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double y(double k), вычисляющую значение *y* для заданного k:

$$y = \begin{cases} k^2 + 3k + 9, \text{если } k \le 3 \\ \dfrac{\sin k}{k^2 - 9}, \text{если } k > 3 \end{cases}.$$

2. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double s(int n), вычисляющую значение s для заданного n:

$$s = \sum_{k=1}^{n} \frac{1}{k^2}.$$

3. Добавьте в проект Calc обработчики сообщений, вызывающие из библиотеки MyLib.dll функции double y(double k) и double s(int n):




*Вариант 5*

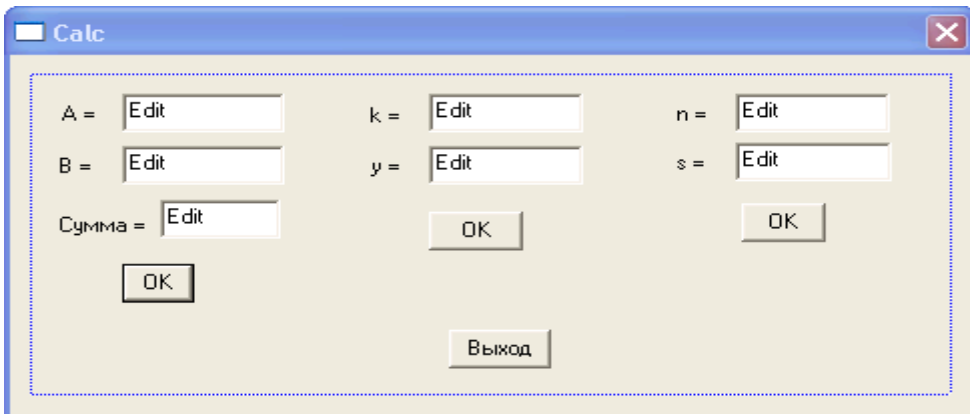
1. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double y(double k), вычисляющую значение *y* для заданного k:

$$y = \begin{cases} \ln k + 9, \text{если } k > 0 \\ -\dfrac{k}{k^2 - 7}, \text{если } k \le 0 \end{cases}.$$

2. Добавьте в библиотеку динамической компоновки MyLib.dll экспортируемую функцию double s(int n), вычисляющую значение s для заданного n:

$$s = \sum_{k=1}^{n} \frac{k}{2 + k}.$$

3. Добавьте в проект Calc обработчики сообщений, вызывающие из библиотеки MyLib.dll функции double y(double k) и double s(int n):




**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ**
**ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ»**

**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 (230100.62) «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**Квалификация (степень) – бакалавр**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Т.М. Афонина
Оператор компьютерной верстки Е.В. Белюсенко

**5700.01.01;МУ.01;2**

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ
# ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ
# ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ»
# НАПРАВЛЕНИЕ ПОДГОТОВКИ»
# 09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

**МОСКВА 2018**

Разработано Е.В. Корнеевой

Под ред. Н.В. Беляниной, к.т.н., доц.

Рекомендовано Учебно-методическим
советом в качестве методических рекомендаций для
педагогических работников и обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ» НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

Методические указания (МУ) подготовлены для педагогических работников и обучающихся, разработаны в соответствии с учебным планом по направлению подготовки 09.03.01 «Информатика и вычислительная техника» на основании требований Федерального государственного образовательного стандарта. МУ предназначены для изучения основ различных технологий программирования и их аспектов в рамках дисциплины «Технология программирования».

**О Г Л А В Л Е Н И Е**

## ВВЕДЕНИЕ

**Цель** лабораторного практикума заключается в изучении основ различных технологий программирования и их аспектов, закреплении теоретических знаний и навыков, полученных на лекциях и практических занятиях.

***Особенность*** данного вида занятий заключается в последовательности осуществления практических и познавательных действий.

Каждое занятие содержит теоретические сведения, практические задания, методику их выполнения.

***Каждое занятие подразделяется на следующие части:***

***Первая – вступительная.*** Обучаемые знакомятся с темой и целью занятия, перечнем прикладного программного обеспечения для проведения исследования или моделирования.

***Вторая – теоретическая.*** Обучаемые самостоятельно изучают теоретические сведения по теме занятия.

***Третья – практическая.*** Обучаемые самостоятельно выполняют практические задания по теме занятия в соответствии с методикой их выполнения.

***Четвёртая – заключительная.*** Предназначена для подведения итогов, контроля качества усвоения материала. Подводятся итоги занятия, обучаемым выставляются оценки.

## Лабораторный практикум № 1 Разработка графического интерфейса. Разработка диалога на основе экранных форм

***Продолжительность: 90 минут.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** познакомиться с новыми возможностями среды PascalABC, осуществить проектирование интерфейса на основе экранных форм.

***Вводная часть***

*Понятие графического интерфейса*

С точки зрения программного обеспечения в состав интерфейса пользователь-компьютер входят два компонента:

1) процесс диалога, то есть двухсторонний обмен информацией между пользователем и компьютером;
2) процесс ввода – вывода, обеспечивающий прием от пользователя и передачу ему данных через различные физические устройства, например дисплей, клавиатура.

Под *графическим интерфейсом* пользователя подразумевается тип экранного представления, при котором пользователь может выбирать команды, запускать задачи и просматривать списки файлов, указывая на пиктограммы или пункты в списках меню, показанных на экране. Действия могут выполняться с помощью мыши или нажатием клавиш на клавиатуре. Типичным примером графического интерфейса пользователя является Windows.

Для пользователя одним из принципиальных преимуществ работы с Windows является то, что большинство имеющихся приложений выглядят и ведут себя сходным образом. Программы для Windows должны включать в себя:

1) главное меню;
2) инструментальную панель быстрых кнопок;
3) контекстное меню;
4) клавиши быстрого доступа;
5) ярлычки подсказок;
6) файл справки;
7) возможность настройки приложения;
8) средства установки приложения и др.

*Разработка диалога на основе экранных форм*

При проектировании пользовательского интерфейса необходимо определить:

- структуру диалога;
- возможный сценарий развития диалога;
- содержание управляющих сообщений и данных, которыми могут обмениваться человек и приложение;
- визуальные атрибуты отображаемой информации.

Выбор структуры диалога – это первый из этапов, который должен быть выполнен при разработке интерфейса. Известно несколько типов диалога. Основными из них являются:

- диалог типа «вопрос – ответ»;
- диалог на основе меню;
- диалог на основе экранных форм;
- диалог на основе командного языка.

Структура диалога типа «*вопрос – ответ*» основана на аналогии с обычным интервью. Система берет на себя роль интервьюера и получает информацию от пользователя в виде ответов на вопросы. В консольных приложениях такой вид диалога используется достаточно часто.

В случае использования *меню* на экран выводится список возможных действий программного комплекса.

При организации диалога *на основе командного языка* программная система не выводит ничего, кроме постоянной подсказки, которая означает готовность системы к работе. Ответственность за правильность задаваемых команд ложится на пользователя.

Диалог на основе *экранных форм* допускает обработку на одном шаге диалога нескольких ответов. На практике формы используются в основном там, где учет какой-либо деятельности требует ввода достаточно стандартного набора данных. Человек, заполняющий форму, может выбирать последовательность ответов, временно пропускать некоторый вопрос, возвращаться назад для коррекции предыдущего ответа. Часто такой вид диалога используется для заполнения бланков в базах данных.

*Среда разработки приложений PascalABC*

Существуют мощные средства для разработки графического пользовательского интерфейса – эти средства называются RAD (rapid application development – быстрая разработка приложений). К таким средствам относятся: Delphi, Visual Studio и другие.

Вам предлагается воспользоваться средой PascalABC, которая предназначена для обучения программированию. В этой среде можно получить основные навыки по программированию графического интерфейса пользователя.

Процесс создания программы состоит из двух шагов: сначала нужно создать форму программы (диалоговое окно), затем – функции обработки событий. Форма приложения создается путем добавления в форму компонентов и последующей их настройки.

Запустите программу PascalABC (рисунок 1).

Выполните команду Сервис/Создать форму.

В рабочем пространстве программы появится форма с заголовком «Форма1». Правее формы расположена панель с управляющими компонентами, левее – страница свойств компонентов (инспектор объектов). В начале проектирования у нас имеется только один компонент – форма.

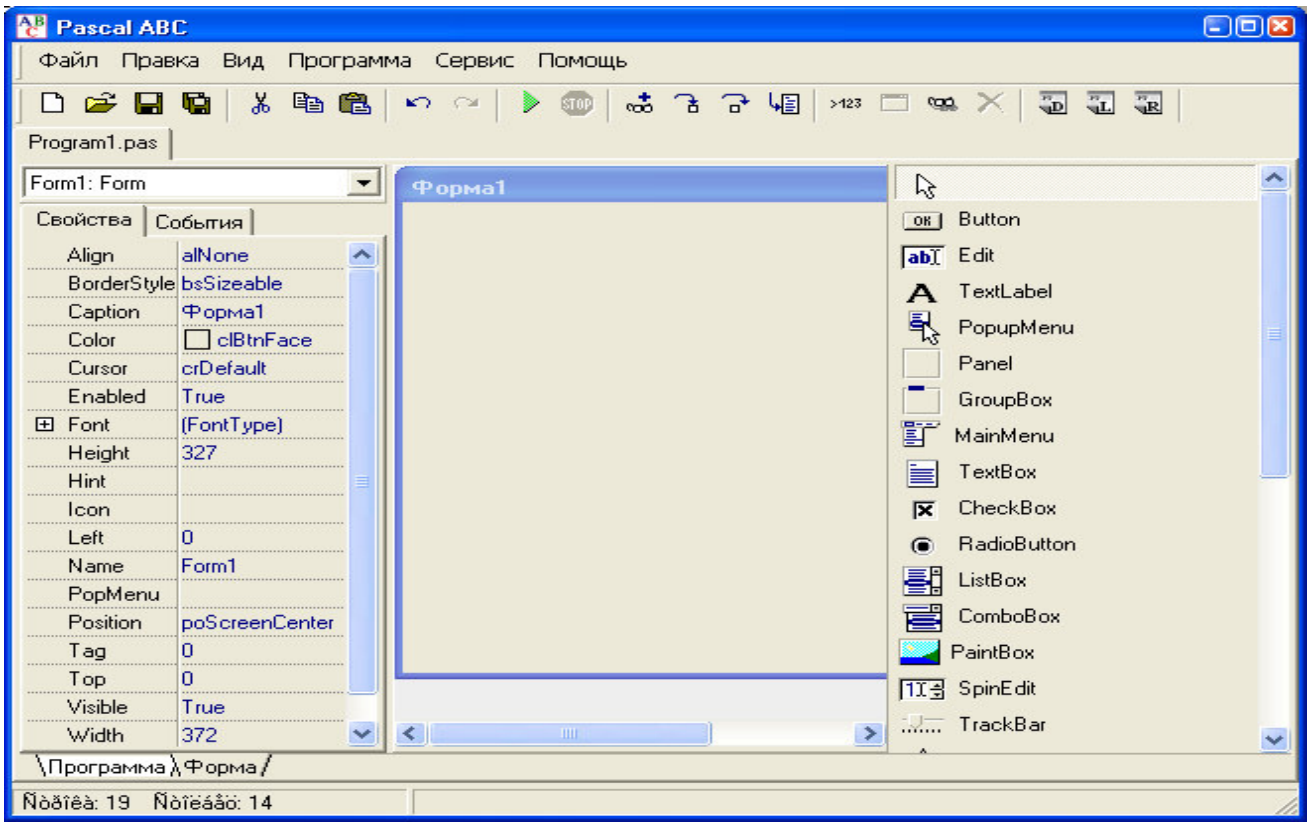



Рисунок 1. Среда разработки PascalABC

В нижней части окна расположены ярлычки страниц «Программа», «Форма». Перейдите в окно программы. Ознакомьтесь с текстом программы.

```
uses vcl;
//$VCLDESIGN+
var
  Form1: Form;
//$VCLDESIGN-
procedure InitControls;// Процедура создания и инициализации формы
begin
  Form1:= Form.Create(0,0,372,327);
  Form1.InitControl(True,False,alNone,crDefault,clBtnFace,'Форма1','');
  Form1.Position:= poScreenCenter;
  Form1.Show;
end;
begin
  InitControls; //Обращение к процедуре
end.
```

Вы видите автоматически сформированный текст процедуры создания и инициализации формы InitControls. С помощью фразы Uses VCL подключается модуль визуальных компонентов.

Для создания экземпляра формы используется конструктор *Create*.

```
constructor Create(L,T,W,H: integer);
```

Параметры `L`, `T`, `W`, `H` задают координаты левого верхнего угла элемента относительно родительского элемента, а также размеры элемента.

Процедура InitControl устанавливает соответствующие свойства элемента.

procedure InitControl(Enabled,Visible: boolean; Align: AlignType; Cursor: CursorType; Color: ColorType; Caption,Hint: string);

Свойство *Position* определяет положение формы относительно экрана. Метод *Show* показывает форму.

При разработке экранной формы нашей задачи воспользуемся визуальными компонентами управления Edit, TextLabel, Button.

*TextLabel* – поле вывода текста. Свойство `Caption` определяет надпись на метке.

*Edit* – поле редактирования текста. Отображает одну строку текста, позволяя ее редактировать. Свойство `Text` определяет текст в поле ввода.

*Button* – командная кнопка. *Caption* – надпись на кнопке.

Метод `SetImage(name: string)` устанавливает рисунок, изображенный на кнопке. В качестве `name` указывается имя файла на диске (допускаются .bmp или .ico файлы) либо имя стандартного ресурса (в этом случае расширение не указывается).

***Практическая часть***

***Разработка интерфейса***

***Задание***

Разработать экранную форму для ввода базы данных «Записная книжка». В базу данных вводятся следующие анкетные данные: фамилия, имя, адрес, телефон, электронный адрес, дата рождения, характер знакомства. Построить приложение для проверки правильности ввода данных.

Выполняем команду *Сервис/Создать* форму и конструируем экранную форму в соответствии с рисунком.

Для формы устанавливаем свойство *BorderStyle* равным *bsDialog*. В этом случае форма будет иметь только одну размерную кнопку – кнопку закрытия окна. В нашей задаче нет необходимости изменять размеры окна, а поэтому не нужно масштабировать размеры элементов управления.

Для изменения заголовка окна используем свойство формы *Caption*.

Окна ввода объединяем в группы с помощью контейнеров *Panel* (на рисунке не обозначены).

Изменяем свойство *Caption* для всех меток, установленных на форму (рисунок 2). Это же свойство изменяем и у кнопки Button1.

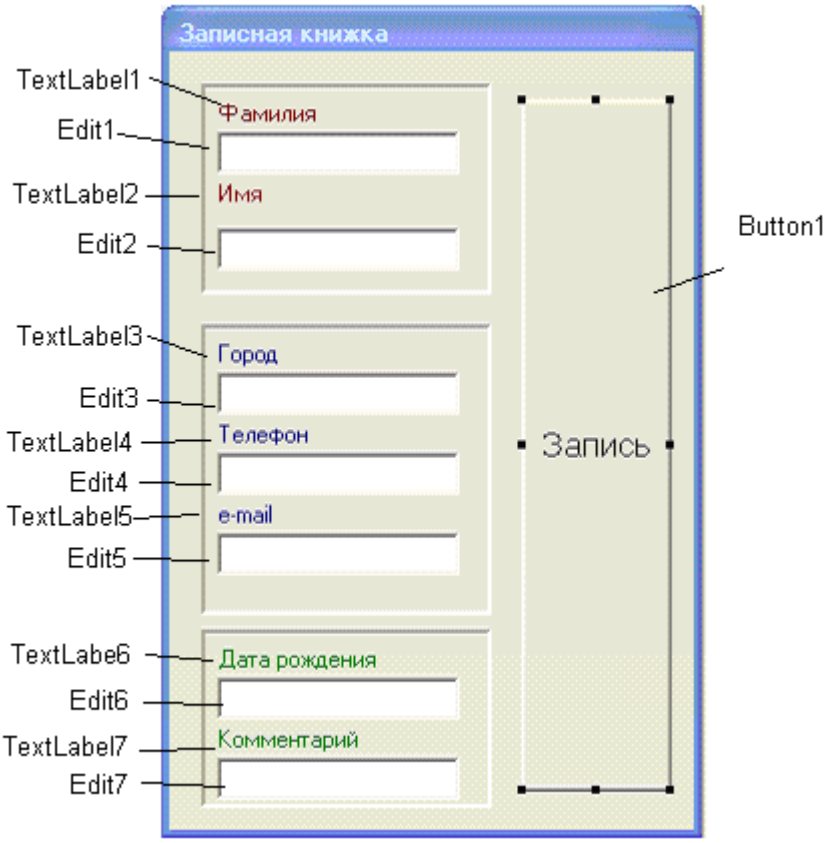



Рисунок 2. Экранная форма для решения поставленной задачи

***Разработка функциональности приложения***

Основную работу в программе выполняют функции обработки событий. Функциональность программы определяется совокупностью ее реакций на те или иные события. В связи с этим каждый компонент помимо свойств характеризуется также набором событий, на которые он может реагировать. События, на которые могут реагировать компоненты, также перечислены в инспекторе объектов (закладка события). Чтобы заставить программу реагировать на событие, необходимо написать фрагмент программы, который называется обработчиком событий.

Для сохранения вводимых данных воспользуемся типизированным файлом.

Тип компоненты файла объявим глобально. Тип данных anketa является записным типом с полями:

name1 – фамилия;

name2 – имя;

Adress – адрес;

Tel – телефон;

Email – электронный адрес;

Bday – дата рождения;

Comm – характер знакомства.

Для упрощения примера сделаем все поля строковыми – String [20].

Файловая переменная F также описывается глобально, так как она будет использоваться в трех обработчиках события.

Часть кода программы, сформированная вручную:

```
Type anketa = record
            name1, name2, Adress, Tel, Email, Bday, Comm: String [20];
          end;
Var f: File of anketa;
// Событие OnActivate для формы возникает при  активизации формы
procedure Form1OnActivate;
begin
assign(f,'Proba'); // Связываем файловую переменную с файлом на диске
ReWrite(f); // Создаем новый файл на диске
end;
// OnClose возникает при закрытии формы
procedure Form1OnClose;
begin
  close(f); // Файл закрывается при окончании работы с формой
end;
procedure Button1OnClick;
Var z:anketa;
begin
 With z do
  Begin
   Name1 := Edit1.text; // Присваиваем содержимое редакторов полям записи
   Name2 := Edit2.text;
   Adress := Edit3.text;
   Tel :=  Edit4.text;
   Email :=  Edit5.text;
   Bday :=  Edit6.text;
   Comm :=  Edit7.text;
 End;
```

```
 Write(f, z); // Записываем компоненту файла на диск
// Очищаем содержимое редакторов
  Edit1.text :='';Edit2.text :='';Edit3.text :='';Edit4.text :='';
  Edit5.text :='';Edit6.text :='';Edit7.text :='';
end;
```

*Разработка приложения для проверки правильности ввода информации*

После того, как программа будет протестирована, необходимо осуществить проверку правильности ввода. Для этого необходимо создать форму. аналогичную предыдущей (рисунок 3).

При определении функциональности формы написать код считывания информации из файла и отображении этой информации в форме.

*Код программы:*

```
Type anketa = record
              name1, name2, Adress, Tel, Email, Bday, Comm: String [20];
            end;
Var f: File of anketa;
procedure Form1OnActivate;
begin
assign(f,'Proba'); Reset(f); // Открываем файл на чтение
```

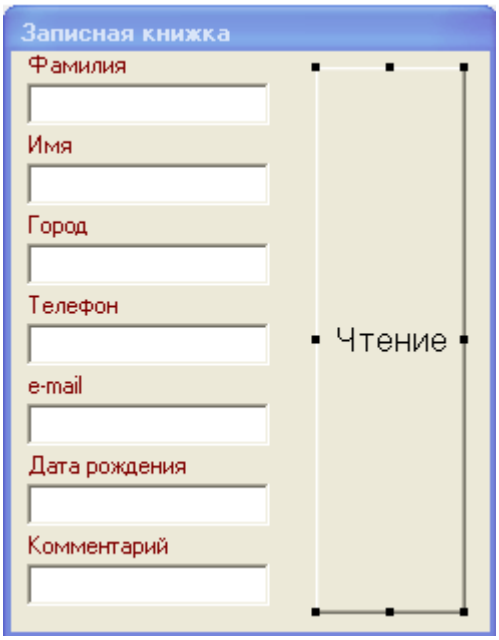



Рисунок 3. Экранная форма для проверки правильности ввода

```
end;
procedure Form1OnClose;
begin
  close(f);
end;
procedure Button1OnClick;
Var z:anketa;
begin
Read(f,z); // Считываем компоненту файла с диска
 With z do
  Begin
    Edit1.text := Name1; // Отображаем информацию в окне редактора
```

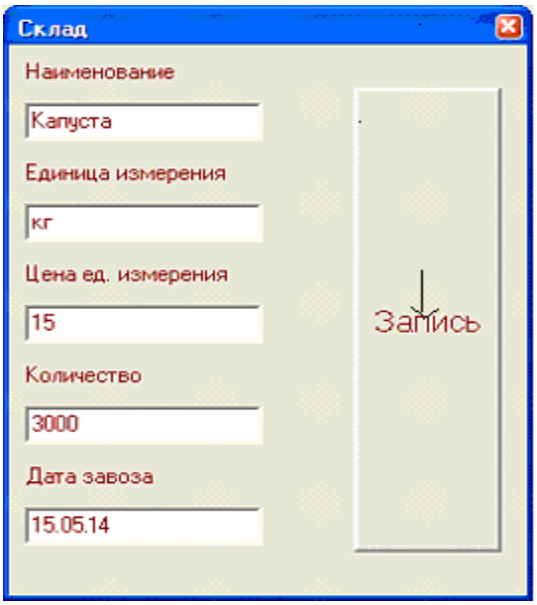
```
   Edit2.text := Name2;
   Edit3.text := Adress;
   Edit4.text := Tel;
   Edit5.text := Email;
   Edit6.text := Bday;
   Edit7.text := Comm;
 End;
end;
```

***Самостоятельная работа***

*Вариант 1*

Разработать приложение для Windows, использующее для ввода экранную форму. Организовать ввод данных в базу «Склад». База товаров, хранящихся на складе: наименование, единица измерения, цена единицы измерения, количество, дата завоза.

1. Разработайте интерфейс с помощью компонентов PascalABC.



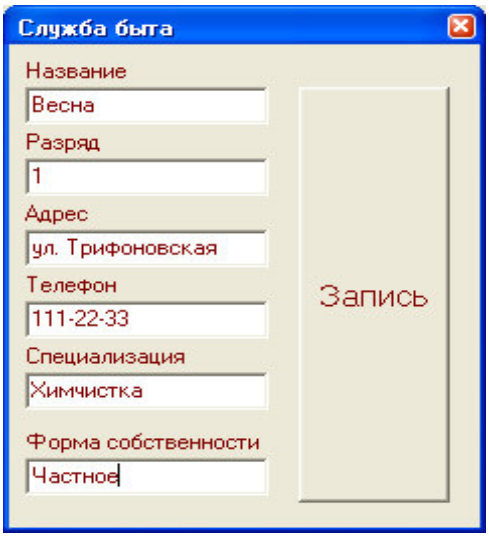

2. Разработайте функциональность приложения.

3. Организуйте проверку правильности ввода исходной информации.

*Вариант 2*

Разработать приложение для Windows, использующее для ввода экранную форму. Организовать ввод данных в базу «Служба быта». База предприятий бытового обслуживания: название, разряд, адрес, телефон, специализация, форма собственности.

1. Разработайте интерфейс с помощью компонентов PascalABC.




2. Разработайте функциональность приложения.

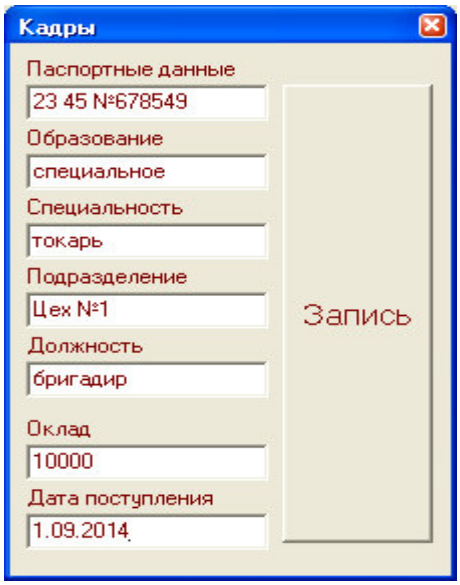
3. Организуйте проверку правильности ввода исходной информации.

*Вариант 3*

Разработать приложение для Windows, использующее для ввода экранную форму.

Организовать ввод данных в базу «Отдел кадров». База данных о сотрудниках фирмы: паспортные данные, образование, специальность, подразделение, должность, оклад, дата поступления в фирму.

1. Разработайте интерфейс с помощью компонентов PascalABC.




2. Разработайте функциональность приложения.

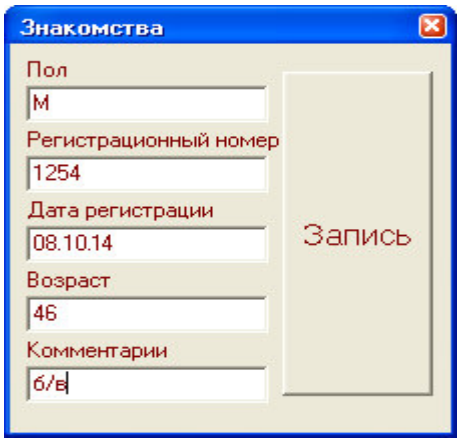
3. Организуйте проверку правильности ввода исходной информации.

*Вариант 4*

Разработать приложение для Windows, использующее для ввода экранную форму.

Организовать ввод данных в базу «Бюро знакомств». База потенциальных женихов и невест: пол, регистрационный номер, дата регистрации, дата рождения, требования к партнеру.

1. Разработайте интерфейс с помощью компонентов PascalABC.




2. Разработайте функциональность приложения.

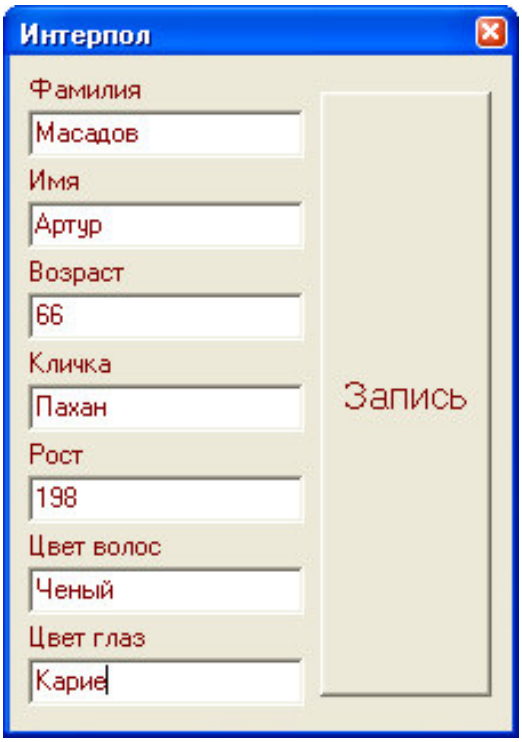
3. Организуйте проверку правильности ввода исходной информации.

*Вариант 5*

Разработать приложение для Windows, использующее для ввода экранную форму.

Организовать ввод данных в базу «Интерпол». Данные по каждому зарегистрированному преступнику: фамилия, имя, возраст, кличка, рост, цвет волос, цвет глаз, особые приметы.

1. Разработайте интерфейс с помощью компонентов PascalABC.




2. Разработайте функциональность приложения.

3. Организуйте проверку правильности ввода исходной информации.

**Лабораторный практикум № 2 Разработка графического интерфейса. Разработка интерфейса на основе меню**

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 1.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** научиться разрабатывать интерфейс на основе меню.

***Вводная часть***

*Диалог на основе меню*

Меню является наиболее популярным вариантом организации запросов на ввод данных во время диалога, управляемого компьютером. Существует несколько основных форматов представления меню на экране:

- список объектов, выбираемых прямым указанием (указанием номера);
- меню в виде блока данных;
- меню в виде строки данных;
- меню в виде пиктограмм.

Меню в виде строки данных может появляться вверху или внизу экрана и часто остается в этой позиции на протяжении всего диалога.

Меню в виде пиктограмм представляет собой множество объектов выбора, разбросанных по всему экрану. Панели инструментов в Windows – пример такого меню.

Меню – это наиболее удобная структура диалога для неподготовленных пользователей.

*Выбор объектов прямым указанием*

Выбор является основным средством, с помощью которого пользователь идентифицирует интересующие его объекты. Реализация модели взаимодействия, основанной на использовании выбора, – один из наиболее важных аспектов проектирования интерфейса. Выбор предполагает прямое указание пользователем идентифицируемого объекта. Этот механизм известен как явный выбор. Если объект выбран, пользователь может определить действие для него.

В средах быстрой разработки приложений существуют компоненты, которые отображают списки строк и позволяют выбрать в них нужную строку. К таким компонентам относятся: *ListBox, CheckListBox, ValueListEditor, ComboBox.*

PascalABC предоставляет возможность использования компонентов *ListBox*, *ComboBox*. Основное свойство обоих компонентов, содержащее список строк, – *Items*, имеющее тип TStrings. Заполнить его во время проектирования можно, нажав кнопку с многоточием около этого свойства в окне Инспектора Объектов. Свойство *ItemsCount* определяет количество строк в списке. *ItemIndex* определяет номер текущего выбранного элемента в списке *Items* (нумерация начинается с 0). При изменении значения *ItemIndex* выбирается строка списка с соответствующим номером.

*ComboBox* – комбинированный список выбора. Свойство *Text* определяет текст в поле ввода. Свойства *Items, ItemsCount, ItemIndex* имеют те же значения, что и для компонента ListBox.

Для выбора какого-либо численного значения можно использовать такие компоненты, как *TrackBar и SpinEdit*.

Класс *SpinEdit* представляет поле для ввода целого значения, совмещенное с двумя кнопками, позволяющими увеличивать и уменьшать это значение. Следующие свойства определены в классе SpinEdit:

*Value* – задает текущее целое значение, содержащееся в компоненте.

*Min* – задает минимальное целое значение, которое пользователь может ввести в компоненте.

*Max* – задает максимальное целое значение, которое пользователь может ввести в компоненте.

Класс *TrackBar* представляет стандартную линейку значений с бегунком. Этот компонент предназначен для визуального управления числовой величиной. Свойства, определенные в классе TrackBar:

*Min* – определяет наименьшее значение бегунка.

*Max* – определяет наибольшее значение бегунка.

*Position* – определяет текущее значение бегунка.

*Frequency* – определяет интервал между делениями, отображаемыми на элементе.

*Orientation* – определяет ориентацию элемента.

*MarkerSize* – определяет размер бегунка.

*TickMarks* – определяет, как отображать деления на бегунке. Тип TickMarksType определен следующим образом: TickMarksType=(tmBottomRight, tmTopLeft, tmBoth).

***Практическая часть***

*Разбор примера разработки приложения*

Этап 1

Условие задачи

Создать приложение для Windows, которое представляет собой элементарный калькулятор, выполняющий четыре арифметических действия: сложение, вычитание, умножение, деление.

Разработка интерфейса (рисунок 4).

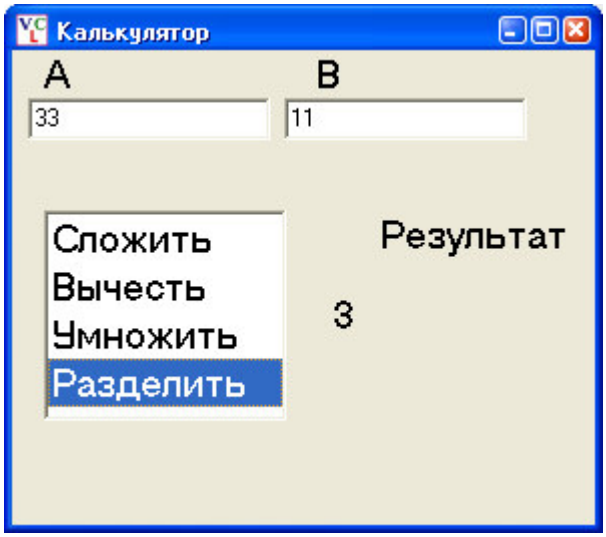



Рисунок 4. Интерфейс приложения

*Форма для решения задачи. Этап 1.*

Для ввода исходной информации используются два текстовых редактора *Edit1* и *Edit2*. Для выбора арифметического действия используется компонент *ListBox1*. Для вывода результата используется метка *TextLabel2*.

Класс *ListBox* представляет элемент управления списка. Свойство *Items* представляет список строк, отображаемый элементом *ListBox*. Список заполняется в Инспекторе объектов.

Функциональность формы определяется обработчиком события ListBox1OnClick. Делаем двойной щелчок по компоненту для создания заготовки обработчика события. При выборе арифметического действия из списка выполняется следующая процедура.

Программа:

```
procedure ListBox1OnClick;
Var a, b, c: real;
begin
a:=StrToFloat(edit1.text); //Преобразование текстовой переменной свойства Text в тип real
b:=StrToFloat(edit2.text);
  Case ListBox1.itemindex of //В зависимости от выбранной строки списка
    0: c:=a+b;
    1: c:=a-b;
    2: c:=a*b;
    3: begin
if b=0 Then //Если второй операнд 0, то деление не производить
      begin
        TextLabel4.caption := 'Деление на 0';
```

```
        exit;
      end;
c:=a/b;
      end;
  end;
  Textlabel4.caption := FloatToStr(c); // Результат отобразить в метке
end;
```

*Этап 2*

Добавить в приложение компоненты TrackBar и SpinEdit для визуального управления численным значением содержимого редакторов, т.е. значениями A и B, которые являются операндами в арифметических действиях.

*Разработка формы*

Устанавливаем на форму компонент SpinEdit1 для определения численного значения операнда A. Задаем свойства компонента: Min = -100; Max = 100.

Добавляем на форму компонент TrackBar1. Задаем свойства компонента: Min = -100; Max = 100.

Аналогично добавляем компоненты SpinEdit2 и TrackBar2 для установки численного значения операнда B (рисунок 5).

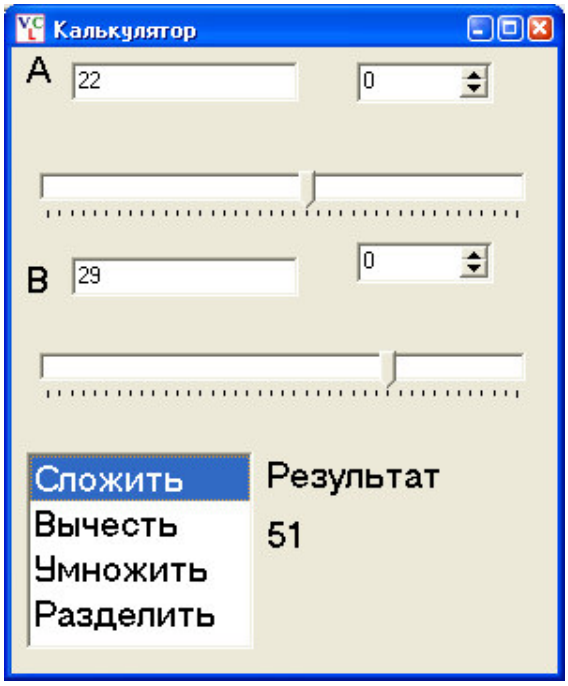



Рисунок 5. Использование элемента управления TrackBar

*Форма для решения задачи. Этап 2.*

Для работы с этими компонентами (SpinEdit, TrackBar) определяем обработчики событий OnChange. В процедурах связываем свойство Position для TrackBar со свойством Text компонента Edit.

Для компонента SpinEdit связываем свойство *Value* со свойством *Text* компонента Edit.

```
procedure TrackBar1OnChange;
begin
  Edit1.text := FloatToStr(TrackBar1.Position); // Операнд A
end;
procedure TrackBar2OnChange;
begin
```

```
  Edit2.text := FloatToStr(TrackBar2.Position); // Операнд B
end;
procedure SpinEdit1OnChange;
begin
    Edit1.text := FloatToStr(SpinEdit1.Value); // Операнд A
end;
procedure SpinEdit2OnChange;
begin
   Edit2.text := FloatToStr(SpinEdit2.Value); // Операнд B
end;
```

***Самостоятельная работа***

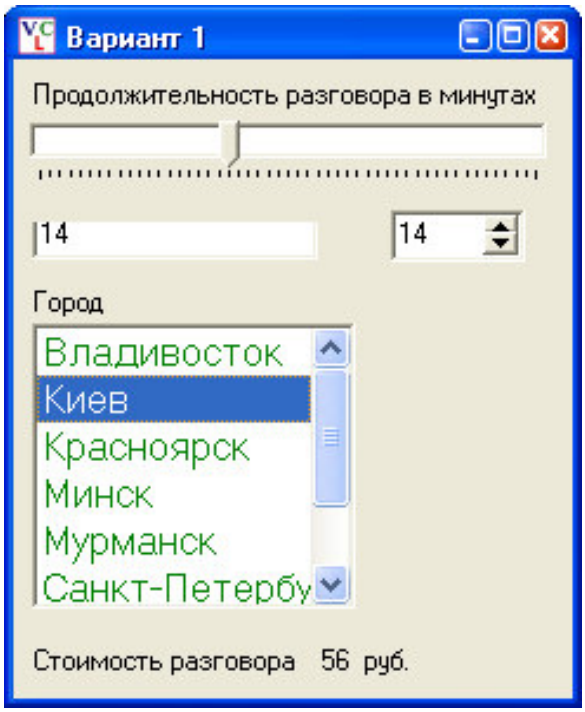
*Вариант 1*

*Условие*

Вычислить стоимость междугороднего телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент, – является константой). Исходными данными для программы являются код города и продолжительность разговора.

Разработать приложение для Windows, используя базовые элементы управления (ListBox, ComboBox, SpinEdit, TrackBar).

1. Разработать графический интерфейс.




2. Определить функциональность формы.

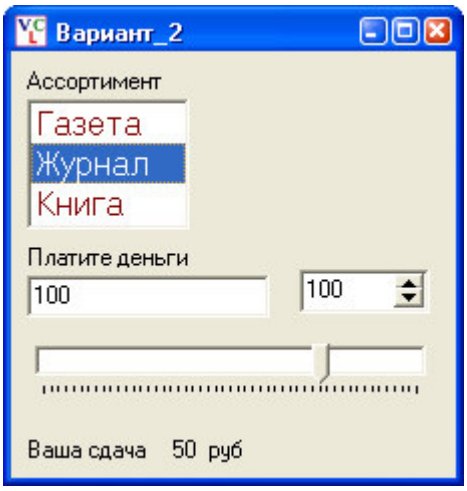
*Вариант 2*

*Условие*

В киоске продаются газета стоимостью 10 рублей, журнал стоимостью 50 рублей и книга стоимостью 140 рублей. Составить программу, которая дает возможность выбрать покупку по желанию (газета, журнал, книга), принимает деньги и выдает причитающуюся сдачу.

Разработать приложение для Windows, используя базовые элементы управления (ListBox, ComboBox, SpinEdit, TrackBar).

1. Разработать графический интерфейс.




2. Определить функциональность формы.

*Вариант 3*

*Условие*

Осуществить по выбору перевод единиц измерения длины из метров в дюймы, ярды, версты, сажень.

1 дюйм = 2,54 см.

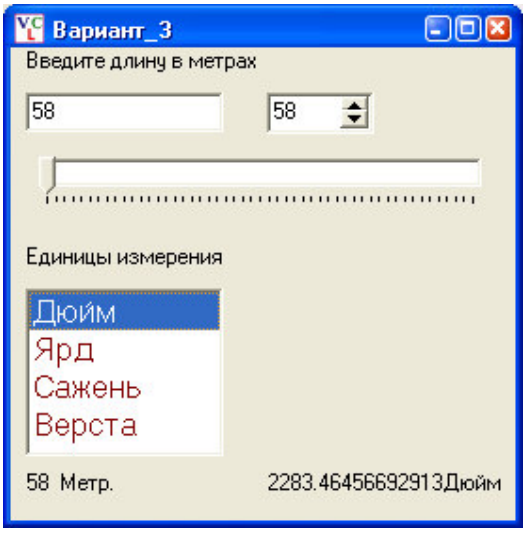
1 ярд = 0,9144 м.

1 верста = 500 сажень.

1 косая сажень = 1,76 м.

Разработать приложение для Windows, используя базовые элементы управления (ListBox, ComboBox, SpinEdit, TrackBar).

1. Разработать графический интерфейс.




2. Определить функциональность формы.

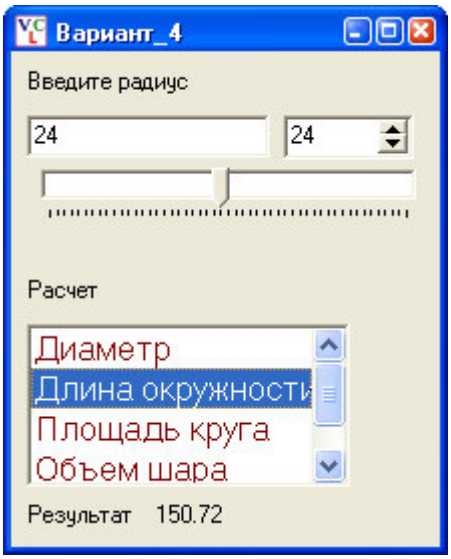
*Вариант 4*

*Условие*

Дан радиус R. Осуществить расчет по выбору: диаметра, длины окружности, площади круга, объема шара.

Разработать приложение для Windows, используя базовые элементы управления (ListBox, ComboBox, SpinEdit, TrackBar).

1. Разработать графический интерфейс.




2. Определить функциональность формы.

*Вариант 5*

*Условие*

Дано: сумма денег в рублях. Осуществить перевод денег в иностранную валюту: доллары, евро, франки, йены.

1$ = 34 руб.

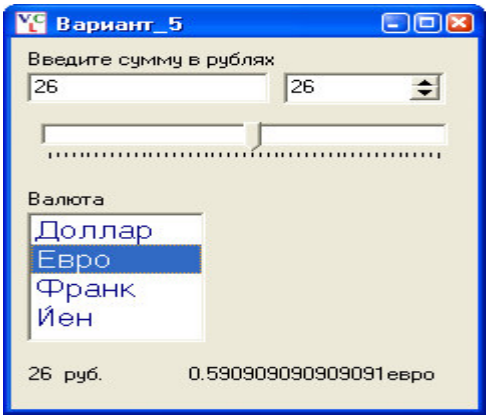
1€ = 44 руб.

USD/CHF = 1,16 (доллар к швейцарскому франку).

USD/YEN = 100 (доллар к японской йене).

Разработать приложение для Windows, используя базовые элементы управления (ListBox, ComboBox, SpinEdit, TrackBar).

1. Разработать графический интерфейс.




2. Определить функциональность формы.

**Лабораторный практикум № 3 Разработка графического интерфейса. Разработка главного и вспомогательного меню**

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** научиться разрабатывать интерфейс на основе меню.

***Вводная часть***

*Стиль меню*

*Меню в виде строки* данных может появляться вверху или внизу экрана и часто остается в этой позиции на протяжении всего диалога. Посредством меню удобно отображать возможные варианты данных для ввода, доступных в любое время работы с системой. Если в системе имеется достаточно большое разнообразие вариантов действий, организуется иерархическая структура из соответствующих меню. Дополнительные меню в виде блоков данных «всплывают» на экране в позиции, определяемой текущим положением указателя, либо «выпадают» непосредственно из строки верхнего уровня. Эти меню исчезают после выбора варианта.

В таком виде выполнено главное меню окна, расположенное непосредственно под полосой заголовка первичного окна. Главное меню называют также *полосой меню*. Полоса меню содержит названия пунктов меню, каждый из которых предоставляет доступ к выпадающему меню.

Содержание главного меню и связанных с ним выпадающих меню определяется функциональным предназначением создаваемого приложения и контекстом выполняемого задания.

Основное требование к меню – их стандартизация. Это требование относится ко многим аспектам меню: месту размещения заголовков меню и их разделов, форме самих заголовков, клавишам быстрого доступа, организации каскадных меню. Цель стандартизации – облегчить пользователю работу с приложением. Надо, чтобы пользователю не приходилось думать, в каком меню и как ему надо открыть или сохранить файл, как получить справку, как работать с буфером обмена и т.д. Для осуществления всех этих операций у пользователя, поработавшего хотя бы с несколькими приложениями Windows, вырабатывается стойкий автоматизм действий, и недопустимо этот автоматизм ломать.

Состав меню зависит от конкретного приложения, но размещение общедоступных разделов должно быть стандартным. Все пользователи уже привыкли, что меню «Файл» размещается слева в полосе главного меню, раздел справки – справа, перед ним в приложениях MDI размещается меню «Окно» и т.д. Главное меню должно также снабжаться инструментальной панелью, быстрые кнопки которой дублируют наиболее часто используемые команды меню.

По возможности стандартным должно быть и расположение разделов и выпадающих меню.

Названия разделов меню должны быть привычными для пользователя. Если Вы не знаете, как назвать какой-то раздел, не изобретайте свое имя, а попытайтесь найти аналогичный раздел в какой-нибудь русифицированной программе OpenOffice. Названия должны быть краткими и понятными. Не используйте фразы, да и вообще более двух слов, поскольку это перегружает экран и замедляет выбор пользователя. Названия разделов должны начинаться с заглавной буквы.

Названия разделов меню, связанных с вызовом диалоговых окон, должны заканчиваться многоточием, показывающим пользователю, что при выборе этого раздела ему предстоит установить в диалоге еще какие-то параметры.

Разделы, к которым относятся каскадные меню, должны заканчиваться стрелкой, указывающей на наличие дочернего меню данного раздела. Злоупотреблять каскадным меню не следует, так как пользователю не так просто до них добираться. Если в дочернем меню должно быть много разделов, например, связанных с какими-то опциями и настройками, то подумайте, не лучше ли вместо этого дочернего меню предусмотреть диалоговое окно, в котором эти опции будут более обозримыми и доступными.

В каждом названии раздела должен быть выделен подчеркиванием символ, соответствующий клавише быстрого доступа к разделу. Вряд ли такими клавишами часто пользуются, но традиция указания таких клавиш незыблема.

Многим разделам могут быть поставлены в соответствие горячие клавиши, позволяющие обратиться к команде данного раздела, даже не заходя в меню. Комбинации таких горячих клавиш должны быть традиционными. Например, команды вырезания, копирования и вставки фрагментов текста практически всегда имеют горячие клавиши Ctrl-X, Ctrl-C и Ctrl-V, соответственно.

Многие разделы меню желательно снабжать пиктограммами, причем пиктограммы для стандартных разделов должны быть общепринятыми, знакомыми пользователю.

Не все разделы меню имеют смысл в любой момент работы пользователя с приложением. Например, если в приложении не открыт ни один документ, то бессмысленно выполнять команды редактирования в меню «Правка». Если в тексте документа ничего не изменялось, то бессмысленным является раздел этого меню «Отменить», отменяющий последнюю команду редактирования. Такие меню и отдельные разделы должны делаться временно недоступными или невидимыми.

*Проектирование главного меню окна*

Проектирование меню – не очень быстрый процесс, и обидно повторять его снова и снова для каждого нового приложения, тем более что требование стандартизации приводит к тому, что одни и те же разделы с одинаковыми свойствами кочуют из приложения в приложение.

Для разработки главного меню предназначен компонент *MainMenu*. Этот компонент определяет главное меню формы. На форму можно поместить сколько угодно объектов этого класса, но отображаться в полосе меню в верхней части формы будет только тот из них, который указан в свойстве *Menu* формы.

Это невизуальный компонент, т.е. место его размещения на форме в процессе проектирования не имеет никакого значения для пользователя – он все равно увидит не сам компонент, а только меню, сгенерированное им.

Основное свойство компонента – *Items*. Его заполнение производится с помощью Конструктора Меню, вызываемого двойным щелчком на компоненте *MainMenu* или нажатием кнопки с многоточием рядом со свойством *Items* в окне Инспектора Объектов. При работе в Конструкторе Меню новые разделы можно вводить, помещая курсор в рамку из точек, обозначающую место расположения нового раздела.

В PascalABC у пунктов меню в Инспекторе Объектов нет такого количества свойств и событий, как, например, в Delphi.

Основное событие раздела – *OnClick*, возникающее при щелчке пользователя на разделе. В обработчике этого события надо написать операторы, которые реализуют задуманные Вами действия.

Свойство *Caption* обозначает надпись раздела.

Свойство `Items` определяет массив пунктов меню (индексация производится с единицы).

Свойство `ItemsCount` определяет количество пунктов меню.

Конструктор `Create` создает главное меню и привязывает его к главной форме приложения (т.е. форме, создаваемой первой). Если главная форма не создана, генерируется ошибка.

Коструктор `Create(owner: Form)` – создает главное меню и привязывает его к форме `owner`.

Процедура `Add(caption: string; onClick: procedure)` добавляет пункт меню с заголовком `caption` и обработчиком `onClick`.

Контекстное меню привязано к конкретным компонентам. Оно всплывает, если во время, когда данный компонент в фокусе, пользователь щелкает правой клавишей мыши. Обычно в контекстное меню включают те команды главного меню, которые в первую очередь могут потребоваться при работе с данным компонентом.

Контекстному меню соответствует компонент *PopupMenu*. Поскольку в приложении может быть несколько контекстных меню, то и компонентов *PopupMenu* может быть несколько. У каждого компонента есть свойство *PopupMenu*, которое связывает компонент и контекстное меню этого компонента.

Формирование контекстного всплывающего меню производится с помощью Конструктора Меню, вызываемого двойным щелчком на *PopupMenu*, точно так же, как и для главного меню.

Меню можно создать динамически в процессе выполнения программы. Для добавления пунктов меню используется метод *Add*. Пример создания меню:

```
//Создание меню
uses vcl;
```

```
var MainForm:Form;
procedure FClose;
begin
  MainForm.Close;
end;
begin
  MainForm:=Form.Create(200,200,300,300);          // Создание формы
  with MainForm do begin
    Menu:=MainMenu.Create;                         // Создание меню
                                    //создаем корневые пункты меню
    Menu.Add('Файл');  //1
    Menu.Add('Меню1'); //2
    Menu.Add('Меню2'); //3
                                                   //создаем подменю
    Menu[1].Add('Выход',FClose,'exit');
    Menu[2].Add('Пункт меню');       // ~ Menu.items[2].add(...)
    Menu[2].Add('Подменю');
    Menu[2][2].Add('Пункт подменю1'); // ~ Menu.items[2].items[2].add(...)
    Menu[2][2].Add('Пункт подменю2');
  end;
end.
```

*Стандартные диалоги*

В состав Windows входит ряд типовых диалоговых окон, таких как окно выбора загружаемого файла, окно выбора шрифта, окно для настройки принтера и т.д. Работа со стандартными диалоговыми окнами осуществляется в три этапа.

Вначале на форму помещается соответствующий компонент и осуществляется настройка его свойств. На втором этапе осуществляется вызов стандартного для диалогов метода Execute, который создает и показывает на экране диалоговое окно. Вызов этого метода обычно располагается внутри обработчика какого-либо события. Например, обработчик выбора опции меню «Открыть файл» может вызвать метод execute диалога *TOpenDialog*, обработчик нажатия инструментальной кнопки «сохранить» может вызвать такой же метод у компонента *TSaveDialog* и т.д. Только после обращения к *Execute* на экране появляется соответствующее диалоговое окно. Окно диалога является модальным окном, поэтому сразу после обращения к E*xecute* дальнейшее выполнение программы приостанавливается до тех пор, пока пользователь не закроет окно.

*TOpenDialog* – диалог открытия файла. Свойство *FileName* содержит выбранный файл при успешном завершении диалога.

*TSaveDialog* – диалог сохранения файла. Свойство *FileName* имеет то же самое значение.

*TFontDialog* – создает и обслуживает стандартное окно выбора шрифта. Результат выбора шрифта содержит свойство *Font*.

*TColorDialog* – создает и обслуживает стандартное окно выбора цвета. Свойство *Color* содержит выбранный цвет.

TOpenPictureDialog – специализированный диалог для открытия графических файлов. Свойство FileName содержит имя файла.

TSavePictureDialog – специализированный диалог для сохранения графических файлов.

***Практическая часть***

*Разбор примера разработки приложения*

*Условие задачи*

Создать текстовый редактор на основе компонента TextBox. В приложении предусмотреть главное и контекстное меню. Текстовый редактор должен осуществлять следующие функции: создавать новый файл,

записывать созданный файл на диск, открывать существующий файл, изменять шрифт, выводить форму с информацией о программе. В приложении использовать контекстное меню для изменения шрифта текстового редактора.

*Разработка главного меню*

1. Устанавливаем на форму компонент TextBox. Класс `TextBox` представляет простой текстовый редактор.

Следующие свойства определены в классе `TextBox`:

- `свойство Lines` определяет динамический массив строк текстового редактора;
- `свойство ScrollBars` определяет, имеются ли у редактора полосы прокрутки.
- `свойство ReadOnly` определяет, является ли редактор доступным только для чтения.

2. Проектируем учебное меню вида:

| Файл | Формат | Помощь | Выход |
|---|---|---|---|
| Создать | Шрифт | О программе | |
| Открыть | | | |
| Сохранить | | | |

3. На форму устанавливаем компонент MainMenu и формируем меню согласно условию (рисунок 6).

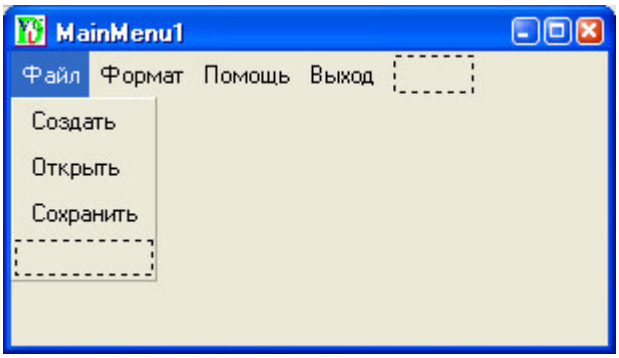



Рисунок 6. Конструктор Меню

Определяем функциональность главного меню формы.

Выбирая щелчком мыши последовательно пункты меню, создаем обработчики событий OnClick (рисунок 7).

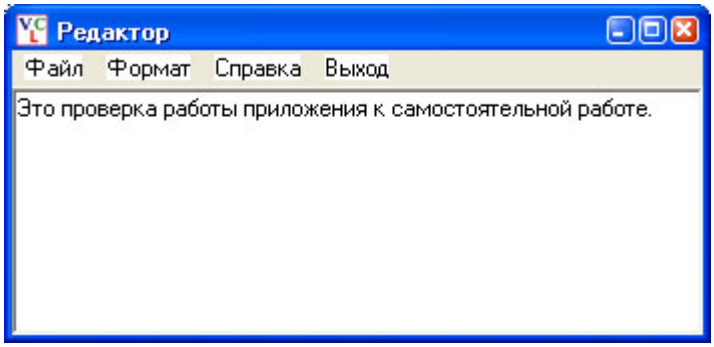



Рисунок 7. Форма приложения

1. Создать новый документ:

```
procedure MainMenu1_1_1OnClick;
begin
```

```
  TextBox1.Clear;                    // Метод Clear очищает редактор
end;
```

2. Открыть существующий документ:

```
procedure MainMenu1_1_2OnClick;
begin
  If OpenDialog1.Execute Then      // Вызываем окно открытия файла
 TextBox1.Lines.LoadFromFile(OpenDialog1.FileName);
end;
```

3. Сохранить документ на диске:

```
procedure MainMenu1_1_3OnClick;
begin
  If SaveDialog1.Execute Then      // Вызываем окно сохранения файла
TextBox1.Lines.SaveToFile(SaveDialog1.FileName);
end;
```

4. Обработчик для пункта меню «Формат/Шрифт».

Поскольку на панели элементов нет компонента FontDialog, создадим этот компонент динамически в процессе выполнения программы.

```
procedure MainMenu1_2_1OnClick;
 Var FontDialog1 : FontDialog;
begin
    FontDialog1 := FontDialog.Create;        // Создаем новый объект
    If FontDialog1.Execute Then              // Вызываем диалоговое окно выбора шрифта
 TextBox1.Font := FontDialog1.Font;
end;
```

5. Обработчик для пункта меню «Помощь/О программе» (рисунок 8).

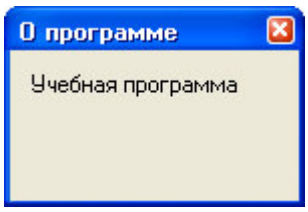



Рисунок 8. Справочное окно «О программе»

В каждом приложении существует такой пункт меню для вывода окна с сообщением об используемой программе, ее разработчиках, версии и другой справочной информации.

Окно с информацией необходимо создать. Для вывода текстовой информации необходимо создать компонент для отображения текста. Это может быть, например, метка.

```
procedure MainMenu1_3_1OnClick;
  Var F1: Form; TextL : TextLabel; // Описываем новые объекты
begin
  F1 := Form.Create(400,300,150,100,'О программе');          //Создаем форму
  F1.borderStyle := bsDialog;                                //Определяем стиль формы
 //F1.Parent := Form1;                    //Определяем родителя созданной формы
 F1.Show;
  Textl := TextLabel.Create(10,10,'Учебная программа'); //Создаем метку
  TextL.Parent := F1;                                 //Определяем родителя созданной метки
end;
```

*Разработка контекстного меню*

Устанавливаем на форму компонент PopupMenu. Двойным щелчком мыши вызываем Конструктор Меню и определяем его пункты (рисунок 9).

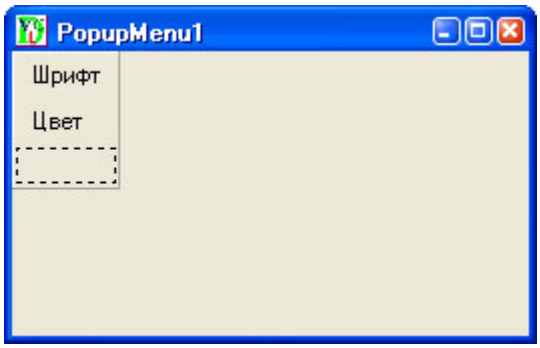



Рисунок 9. Конструктор контекстного Меню

Функциональность контекстного меню определяют два обработчика события OnClick. Поскольку в главном меню есть вызов диалогового окна выбора шрифта, то можно вызвать этот обработчик внутри обработчика пункта контекстного меню.

```
procedure PopupMenu1_1OnClick;
begin
  MainMenu1_2_1OnClick;          // Вызов обработчика главного меню
end;
procedure PopupMenu1_2OnClick;
begin
 If ColorDialog1.Execute Then                //Вызов диалогового окна выбора цвета
 TextBox1.Color := ColorDialog1.Color;
end;
```

***Самостоятельная работа***

*Вариант 1*

*Условие:*

Вычислить стоимость междугороднего телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент, – является константой). Исходными данными для программы являются код города и продолжительность разговора.

1. Разработать главное меню. В пункте меню «Формат» менять шрифт и цвет компонентов ListBox или ComboBox. Структура меню:

Главное меню:

| *Город* | *Формат* | *Помощь* | *Выход* |
|---|---|---|---|
| *Владивосток* | *Шрифт* | *О программе* | |
| *Киев* | *Цвет* | | |
| *Красноярск* | | | |
| *Ярославль* | | | |

2. Разработать контекстное меню.

*Контекстное меню*

*Шрифт* (для метки).

*Цвет* (для формы).

3. В приложении использовать стандартные диалоги.

*Вариант 2*

*Условие*

В киоске продаются газета стоимостью 10 рублей, журнал стоимостью 50 рублей и книга стоимостью 140 рублей. Составить программу, которая дает возможность выбрать покупку по желанию (газета, журнал, книга), принимает деньги и выдает причитающуюся сдачу.

1. Разработать главное меню. В пункте меню «Формат» менять шрифт и цвет компонентов ListBox или ComboBox. Структура меню:

Главное меню:

| *Ассортимент* | *Формат* | *Помощь* | *Выход* |
|---|---|---|---|
| *Газета* | *Шрифт* | *О программе* | |
| *Журнал* | *Цвет* | | |
| *Книга* | | | |

2. Разработать контекстное меню.

Контекстное меню:

*Шрифт* (для редактора).

*Цвет* (для метки).

3. В приложении использовать стандартные диалоги.

*Вариант 3*

*Условие*

Осуществить по выбору перевод единиц измерения длины из метров в дюймы, ярды, версты, сажень.

1 дюйм = 2.54 см.

1 ярд = 0.9144 м.

1 верста = 500 сажень.

1 косая сажень = 1.76 м.

1. Разработать главное меню. В пункте меню «Формат» менять шрифт и цвет компонентов ListBox или ComboBox. Структура меню:

Главное меню:

| *Единицы измерения* | *Формат* | *Помощь* | *Выход* |
|---|---|---|---|
| *Дюйм* | *Шрифт* | *О программе* | |
| *Ярд* | *Цвет* | | |
| *Сажень* | | | |
| *Верста* | | | |

2. Разработать контекстное меню.

Контекстное меню:

*Шрифт* (для метки).

*Цвет* (для формы).

3. В приложении использовать стандартные диалоги.

*Вариант 4*

*Условие*

Дан радиус R. Осуществить расчет по выбору: диаметра, длины окружности, площади круга, объема шара.

1. Разработать главное меню. В пункте меню «Формат» менять шрифт и цвет компонентов ListBox или ComboBox. Структура меню:

Главное меню:

| *Вид расчета* | *Формат* | *Помощь* | *Выход* |
|---|---|---|---|
| *Диаметр* | *Шрифт* | *О программе* | |
| *Длина окружности* | *Цвет* | | |
| *Площадь круга* | | | |
| *Объем шара* | | | |

2. Разработать контекстное меню.

Контекстное меню:

*Шрифт* (для метки).

*Цвет* (для формы).

3. В приложении использовать стандартные диалоги.

*Вариант 5*

*Условие*

Дано: сумма денег в рублях. Осуществить перевод денег в иностранную валюту: доллары, евро, франки, йены.

1$ = 34 руб.

1€ = 44 руб.

USD/CHF = 1,16 (доллар к швейцарскому франку);

USD/YEN = 100 (доллар к японской йене).

1. Разработать главное меню. В пункте меню «Формат» менять шрифт и цвет компонентов ListBox или ComboBox. Структура меню:

Главное меню:

| *Валюта* | *Формат* | *Помощь* | *Выход* |
|---|---|---|---|
| *Доллар* | *Шрифт* | *О программе* | |
| *Евро* | *Цвет* | | |
| *Франк* | | | |
| *Йень* | | | |

2. Разработать контекстное меню.

Контекстное меню:

*Шрифт* (для метки).

*Цвет* (для формы).

3. В разработке использовать стандартные диалоги.

**Лабораторный практикум № 4 Разработка графического интерфейса. Инструментальные панели – элемент графического интерфейса**

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить процесс создания инструментальных панелей и статус-строки в среде PascalABC.

***Вводная часть***

*Создание инструментальной панели на основе компонента ToolBar*

Панели являются контейнерами, служащими для объединения других управляющих элементов. Они могут выполнять как чисто декоративные функции, зрительно объединяя компоненты, связанные друг с другом по назначению, так и функции управления.

В языке PascalABC существуют следующие компоненты для организации панелей:

*GroupBox* – контейнер, объединяющий группу связанных органов управления;

*Panel* – контейнер для группирования органов управления и меньших контейнеров;

*Bevel* – используется для рисования прямоугольной рамки, изображенной как выступающая или утопленная;

*ScrollBox* – используется для создания зон отображения с прокруткой;

*StatusBar* – полоса состояния приложения, при необходимости на нескольких панелях;

*ToolBar* – инструментальная панель для быстрого доступа к часто используемым функциям приложения.

Главное меню должно снабжаться инструментальной панелью, быстрые кнопки которой дублируют наиболее часто используемые команды меню. На этих кнопках надо использовать, по возможности, привычные картинки. Панели могут размещаться в любом месте экрана, иметь любую ориентацию. Доступ к элементам управления, размещенным в панелях инструментов, осуществляется либо с помощью мыши, либо с использованием горячих клавиш. Для элементов управления в панели могут использоваться текстовые метки или всплывающие подсказки.

Класс *ToolBar* представляет панель инструментов. На панели инструментов могут находиться только кнопки *ToolButton*, создаваемые методом *AddButton* или методом *AddSeparator*.

Следующие свойства определены в классе ToolBar:

*Buttons*[i: integer]: ToolButton – определяет массив кнопок на панели инструментов. Индексация начинается с единицы.

*ButtonsCount*: integer – определяет количество кнопок. Свойство доступно только для чтения.

*Flat*: boolean – определяет, содержит ли панель инструментов плоские кнопки (Flat=True) или кнопки, имеющие рамку (Flat=False).

Следующие методы унаследованы от класса Control:

*SetSize*(Width,Height: integer) – определяет размеры панели.

*SetPos*(Left,Top: integer) – определяет положение панели относительно разрабатываемой формы.

Следующие методы определены в классе ToolBar:

*AddSeparator* – добавляет на панель инструментов разделитель.

*AddButton*(name: string) – добавляет на панель инструментов кнопку с указанием имени файла name, содержащего рисунок на кнопке (в качестве name может указываться имя стандартного ресурса).

*AddButton*(name: string; n: integer) – добавляет на панель инструментов кнопку с указанием имени файла name, содержащего набор рисунков; n указывает номер рисунка в наборе.

*AddButton*(onClick: procedure; name: string) – добавляет на панель инструментов кнопку с указанием обработчика события onClick и имени файла name, содержащего рисунок на кнопке (в качестве name может указываться имя стандартного ресурса).

*Панель состояния (статус-строка)*

Интерфейс любой серьезной программы должен включать в себя полосу состояния, используемую для развернутых подсказок и выдачи различной информации пользователю.

Компонент *StatusBar* предназначен для создания панелей состояния, которые обычно располагаются в нижней части основной формы. Компонент может иметь несколько секций.

Следующие свойства определены в классе StatusBar:

*Text*: string – определяет текст в первой панели строки статуса.

*Panels*[i: integer]: StatusPanel – определяет набор панелей для строки статуса. Индексация производится с 1.

*PanelsCount*: integer – определяет количество панелей в строке статуса. Вначале создается одна панель. Свойство предназначено только для чтения.

Следующие методы определены в классе StatusBar:

*Add*(text: string) – добавляет панель с заданным текстом к строке статуса.

*Add*(text: string; width: integer) – добавляет панель с заданным текстом и заданной ширины к строке статуса.

Класс *StatusPanel* представляет панель для строки статуса StatusBar.

Следующие свойства определены в классе StatusPanel:

*Text*: string – определяет текст панели.

*Alignment*: AlignmentType – определяет выравнивание текста на панели. Тип AlignmentType определен следующим образом: AlignmentType=(taLeftJustify, taRightJustify, taCenter).

*Width*: integer – определяет ширину панели.

*Bevel*: BevelType – задает стиль рамки панели.

Панели создаются только неявно методом Add класса StatusBar.

***Практическая часть***

*Разбор примера разработки приложения*

На предыдущем лабораторном практикуме выполнена задача по разработке текстового редактора на основе компонента TextBox с главным и контекстным меню.

Текстовый редактор осуществляет следующие функции:

- создает новый файл;
- записывает созданный файл на диск;
- открывает существующий файл;
- изменяет шрифт;
- изменяет цвет фона;
- выводит форму с информацией о программе.

Перед нами стоит задача дополнить приложение панелью инструментов и строкой статуса. Дополнить приложение Всплывающими подсказками.

Выносим на панель инструментов пиктограммы команд, выполняемых с помощью пунктов главного меню: создать, открыть, шрифт, цвет (рисунок10).

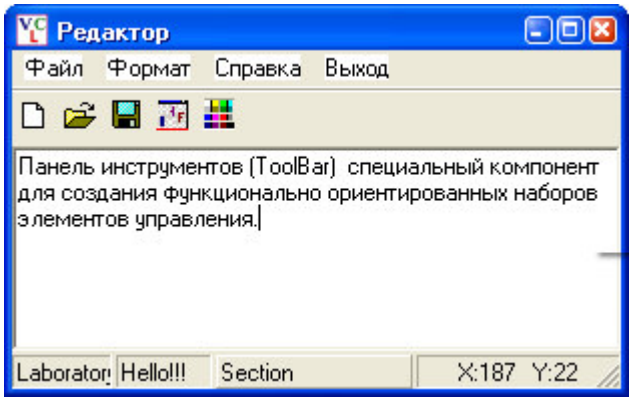



Рисунок 10. Приложение «Редактор» с панелью инструментов и строкой статуса

Компонент для создания панели инструментов ToolBar отсутствует в инспекторе объектов, поэтому его необходимо создать динамически в коде программы.

Добавляем в раздел описания новые объекты:

```
Var FontDialog1 : FontDialog;
  Status : StatusBar; //Описание объекта для строки статуса
  Tb : ToolBar;  // Описание объекта для панели инструментов
```

В обработчик события, происходящим при создании формы, добавляем операторы создания панели и строки статуса:

```
Tb := ToolBar.Create;
Status:=StatusBar.create;
```

Свойству панели Flat присваиваем значение True, что определяет плоские кнопки в панели инструментов.

Кнопки на панель инструментов добавляем методом AddButton, у которого два параметра. Первый – обработчик события, который должен вызываться при нажатии кнопки. Этот обработчик уже создан для соответствующего пункта меню.

Второй параметр – имя файла, содержащего рисунок на кнопке.

Для строки статуса панели создаются с помощью метода Add. Ознакомьтесь с обработчиком события Form1OnCreate. Код обработчика события:

```
procedure Form1OnCreate;
begin
  FontDialog1 := FontDialog.Create;
  Tb := ToolBar.Create; // Создаем панель инструментов
  Tb.Flat := True; // Плоские кнопки на панели
  Tb.AddButton(MainMenu1_1_1OnClick,'New');// Кнопка «Создать»
  Tb.AddButton(MainMenu1_1_2OnClick,'Open');// Кнопка «Открыть»
  Tb.AddButton(MainMenu1_1_3OnClick,'Save'); // Кнопка «Сохранить»
  Tb.AddButton(MainMenu1_2_1OnClick,'FontDialog.bmp'); // Кнопка «Шрифт»
  Tb.AddButton(PopupMenu1_2OnClick,'Color.bmp');              // Кнопка «Цвет»
   Status:=StatusBar.create;                                          // Создаем строку статуса
  Status.caption:='Laboratory n.11';          // Определяем первую секцию строки статуса
  Status.Add('Hello!!!');                     // Определяем вторую секцию строки статуса
  Status.Add('Section');                      // Определяем третью секцию строки статуса
  Status[3].Width:=100;                       // Ширина секции
  Status[3].Bevel:=bvRaised;                        // Задаем стиль рамки
  Status.Add;                                       // Определяем четвертую секцию строки статуса
  Status[4].Text:='Informatika';              // Текст четвертой секции
  Status[4].Aligment:=taCenter;               // Расположение текста по центру
end;
```

В строку состояния можно вывести информацию о текущем состоянии объектов и процессов. Создадим обработчик события, который будет осуществлять вывод координат мыши при щелчке по области текстового редактора. Событие OnMouseDown происходит при щелчке мыши на область текстового редактора. Параметрами обработчика являются X и Y – координаты мыши в этот момент времени. Координаты отображаем в четвертой панели строки статуса.

```
procedure TextBox1OnMouseDown(x,y,button: integer);
begin
 Status[4].Text:='X:'+ IntToStr(x) + '   Y:' + IntToStr(Y);
end;
```

Приложение снабдим всплывающими подсказками. Всплывающая подсказка (Hint) – это небольшое окно, содержащее поясняющий текст, которое появляется на экране, когда пользователь размещает указатель мыши на элементе управления. Подсказка размещается ниже и правее курсора и отображается фиксированное время. Такую подсказку легко организовать с помощью свойства *Hint* компонента.

***Самостоятельная работа***

*Вариант 1*

*Условие*

Вычислить стоимость междугороднего телефонного разговора (цена одной минуты определяется расстоянием до города, в котором находится абонент, – является константой). Исходными данными для программы являются код города и продолжительность разговора.

1. К приложению добавьте панель инструментов, позволяющую: очистить компонент ListBox, загрузить новое содержание в компонент ListBox, сохранить содержимое компонента ListBox. обратиться к пунктам меню «Формат», «Выход».

2. Создайте панель статуса, в которой отобразите Вашу фамилию, а также информацию о текущем положении курсора.

3. Снабдите компоненты приложения всплывающими подсказками.

*Вариант 2*

*Условие*

В киоске продаются газета стоимостью 10 рублей, журнал стоимостью 50 рублей и книга стоимостью 140 рублей. Составить программу, которая дает возможность выбрать покупку по желанию (газета, журнал, книга), принимает деньги и выдает причитающуюся сдачу.

1. К приложению добавить панель инструментов, позволяющую обратиться ко всем пунктам главного меню.

2. Создайте панель статуса, в которой отобразите Вашу фамилию, а также информацию о текущем положении курсора.

3. Снабдите компоненты приложения всплывающими подсказками.

*Вариант 3*

*Условие*

Осуществить по выбору перевод единиц измерения длины из метров в дюймы, ярды, версты, сажень.

1 дюйм = 2.54 см.

1 ярд = 0.9144 м.

1 верста = 500 сажень.

1 косая сажень = 1.76 м.

1. К приложению добавить панель инструментов, позволяющую обратиться ко всем пунктам главного меню.

2. Создайте панель статуса, в которой отобразите Вашу фамилию, а также информацию о текущем положении курсора.

3. Снабдите компоненты приложения всплывающими подсказками.

*Вариант 4*

*Условие*

Дан радиус R. Осуществить расчет по выбору: диаметра, длины окружности, площади круга, объема шара.

1. К приложению добавить панель инструментов, позволяющую обратиться ко всем пунктам главного меню.

2. Создайте панель статуса, в которой отобразите информацию о текущем положении курсора.

3. Снабдите компоненты приложения всплывающими подсказками.

*Вариант 5*

*Условие*

Дано: сумма денег в рублях. Осуществить перевод денег в иностранную валюту: доллары, евро, франки, йены.

1$ = 34 руб.

1€ = 44 руб.

USD/CHF = 1,16 (доллар к швейцарскому франку);

USD/YEN = 100 (доллар к японской йене).

1. К приложению добавить панель инструментов, позволяющую обратиться ко всем пунктам главного меню.

2. Создайте панель статуса, в которой отобразите информацию о текущем положении курсора.

3. Снабдите компоненты приложения всплывающими подсказками.

**Лабораторный практикум № 5 Структурный подход к разработке программного обеспечения. Создание спецификации на разработку программной системы**

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 2.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить на практическом примере процесс создания спецификации на разработку программного комплекса при структурном методе проектирования программ.

***Вводная часть***

*Технологические подходы к разработке программного обеспечения*

Технология проектирования определяется как совокупность трех составляющих:

- пошаговой процедуры, определяющей последовательность технологических операций проектирования;
- критериев и правил, используемых для оценки результатов выполнения технологических операций;
- нотаций (графических и текстовых средств), используемых для описания проектируемой системы.

Существует три основных группы технологических подходов к разработке программного обеспечения:

- подходы со слабой формализацией;
- гибкие подходы;
- строгие подходы.

В первом случае явные технологии не используются. Такой подход уместен только для очень маленьких проектов, как правило, завершающихся созданием демонстрационного прототипа.

Гибкие подходы применяются для небольших или средних проектов в случае неясных или изменяющихся требований к системе. Эти подходы основываются на многократном повторении итераций, в которых участвуют заказчик и разработчик. На каждой итерации программа уточняется и развивается.

Строгие подходы рекомендуется применять для средних, масштабных и гигантских проектов с фиксированным объемом работ. К этой группе относится классический каскадный подход, включающий основные процессы, сложившиеся исторически в результате практического опыта разработки программного обеспечения.

При любом подходе в той или иной комбинации выполняются стадии жизненного цикла программы.

*Жизненный цикл программного обеспечения (ЖЦ ПО)*

Основным нормативным документом, регламентирующим ЖЦ ПО, является международный стандарт ISO/IEC 12207 (ISO – International Organization of Standardization – Международная организация по стандартизации, IEC – International Electrotechnical Commission – Международная комиссия по электротехнике). Он определяет структуру ЖЦ, содержащую процессы, действия и задачи, которые должны быть выполнены во время создания ПО.

Структура ЖЦ ПО по стандарту ISO/IEC 12207 базируется на трех группах процессов:

- основные процессы ЖЦ ПО (приобретение, поставка, разработка, эксплуатация, сопровождение);
- вспомогательные процессы, обеспечивающие выполнение основных процессов (документирование, управление конфигурацией, обеспечение качества, верификация, аттестация, оценка, аудит, решение проблем);
- организационные процессы (управление проектами, создание инфраструктуры проекта, определение, оценка и улучшение самого ЖЦ, обучение).

Жизненный цикл программы – это весь период ее разработки и эксплуатации, начиная с момента возникновения замысла и заканчивая прекращением всех видов ее использования.

В классическом жизненном цикле выделяются следующие процессы.

1. Системный анализ.
2. Анализ требований.
3. Проектирование.
4. Кодирование.
5. Тестирование.
6. Сопровождение.

Результаты каждого этапа отображаются в документах, называемых спецификациями. Первые два этапа жизненного цикла программного обеспечения представляются в виде документа «Постановка задачи». Формы спецификаций могут различаться, но все они имеют несколько обязательных разделов.

***Постановка задачи***

*Сокращенная форма спецификации для раздела «Постановка задачи»*

1. Название задачи.
2. Описание задачи.
3. Управление режимами работы программы.
4. Входные данные.
5. Выходные данные.
6. Возможные ошибки.
7. Контрольные примеры для отладки.

*Название задачи (спецификация)*

Дается краткое определение решаемой задачи, название программного комплекса, указывается система программирования для его реализации и требования к аппаратному обеспечению (компьютеру, внешним устройствам и т.д.).

*Описание*

Подробно излагается постановка задачи, описывается применяемая математическая модель для задач вычислительного характера, метод обработки входных данных для задач не вычислительного (логического) характера и т.д.

*Управление режимами работы программы*

Формулируются основные требования к способу взаимодействия пользователя с программой (интерфейс пользователь–компьютер).

*Входные данные*

Описываются входные данные, указываются пределы, в которых они могут изменяться, значения, которые они не могут принимать, и т.д.

*Выходные данные*

Описываются выходные данные. Указывается, в каком виде они должны быть представлены – в числовом или текстовом, в виде таблицы или Web–документа, на диске, печатающем устройстве или дисплее и другие.

*Возможные ошибки*

Перечисляются возможные ошибки пользователя при работе с программой. Например, ошибки при вводе исходных данных и другие. Указываются способы диагностики (под диагностикой понимается выявление, обнаружение ошибок при работе программного комплекса) и защиты от этих ошибок на этапе проектирования, а также возможная реакция программного комплекса на эти действия.

*Контрольные примеры для отладки*

Приводится один или несколько примеров работы программного комплекса, на которых в простейших случаях проводится его отладка и тестирование.

***Практическая часть***

*Задание на разработку программного обеспечения*

Разработать программный комплекс по обработке результатов сессии на курсе. Программный комплекс позволяет получать следующие сведения:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

*Спецификация*

Название – Деканат.

Название программы – Uchkompl.

Система программирования – PascalABC. Выбор системы программирования обусловлен учебными целями.

Компьютер – IBM PC Pentium.

*Описание*

Необходимо разработать программу, которая дает ответы на следующие вопросы:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

Для выполнения данной задачи в программе должен присутствовать блок ввода исходной информации. Информацию необходимо хранить на диске.

При выполнении запросов исходная информация считывается с диска.

*Запрос 1*

Если хотя бы одна оценка равна двойке, фамилия обучающегося должна быть выведена на экран.

Если $Rez_I = 2$, вывести $F_I$, где I = 1,2,…Kol_Pr.

Kol_Pr – количество предметов, сдаваемых в сессию.

F – фамилии обучающихся.

Rez – оценки по предметам сессии.

I – номер предмета.

*Запрос 2*

Подсчитывается количество обучающихся, сдавших сессию с оценками выше тройки.

Если

$$K = Kol\_Pr,$$

то

$$Ball45 = Ball45 + 1,$$

где

$$K = \sum_{1}^{Kol\_Pr} P10.$$

P10 = 1, если $Rez_i > 3$ и P10 = 0 – в противном случае.

Результат – Ball45/Kol_St *100, где Kol_St – общее количество обучающихся, сдавших сессию.

*Запрос 3*

Рассчитывается средний балл по предмету

$$\mathbf{Sr\_Ball_J} = \frac{\sum_{I=1}^{Kol\_St} Rez_{I,J}}{Kol\_St}.$$

Среди средних значений определяется максимальное значение.

*Запрос 4*

Создается массив групп. Для каждой группы рассчитывается средний балл. Массив средних баллов упорядочивается по убыванию.

*Входные данные*

1. Массив названий предметов, по которым тестировались обучающиеся курса.
2. Информация по каждому обучающемуся курса:

- номер группы;
- фамилия;
- имя;
- отчество;
- оценки по предметам сессии (массив).

*Выходные данные*

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;

- номера групп в порядке убывания средней успеваемости.

Вывод: номер группы, средний балл.

Результаты выводятся на экран. Из эстетических соображений вывод центрируется и обводится рамкой.

*Ошибки*

При вводе данных необходимо предусмотреть интервалы допустимых значений. Оценки должны находиться в интервале от 2 до 5. Это величины целого типа.

Фамилии должны вводиться буквами кириллицы.

Необходима проверка наличия файла на диске для того, чтобы не прервалась работа программы при введении неверного значения.

*Контрольный пример*

Входная информация:

| Номер группы | Фамилия | Имя | Отчество | Физика | Информатика | История |
|---|---|---|---|---|---|---|
| 1 | Иванов | Иван | Иванович | 4 | 3 | 2 |
| 1 | Петрова | Нина | Петровна | 4 | 3 | 3 |
| 1 | Сидоров | Максим | Сергеевич | 4 | 5 | 5 |
| 2 | Грязнова | Ольга | Николаевна | 4 | 4 | 4 |
| 2 | Усова | Ирина | Ивановна | 3 | 4 | 5 |
| 2 | Попов | Матвей | Евгеньевич | 4 | 3 | 2 |
| 2 | Панов | Степан | Валерьевич | 4 | 5 | 5 |
| 3 | Жуков | Михаил | Акимович | 3 | 2 | 2 |
| 3 | Грузь | Тимур | Вепьевич | 4 | 4 | 3 |

*Результаты расчета*

1. Двоечники:

Иванов Иван Иванович.

Попов Матвей Евгеньевич.

Жуков Михаил Акимович.

2. Процент отличников – 33,3 %.

3. Лучший результат по дисциплине – физика.

4. Группы в порядке убывания среднего балла:

группа 2 3,917;

группа 1 3,778;

группа 3 3.

***Самостоятельная работа***

*Вариант 1*

Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота).

1. Создайте спецификацию на разработку программного комплекса. Результаты разработки поместите в текстовый документ. Дайте файловому документу имя «Постановка задачи». Этот файл потребуется Вам для выполнения следующих лабораторных практикумов.

2. Составьте описание возможностей разрабатываемого комплекса. Результаты разработки поместите в текстовый документ «Постановка задачи».

3. Составьте описание входных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

4. Составьте описание выходных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

5. Составьте описание ошибок, которые могут возникнуть при работе с программой. Результаты поместите в текстовый документ «Постановка задачи».

6. Составьте контрольный пример для проверки правильности работы программы. Результаты разработки поместите в текстовый документ «Постановка задачи».

*Вариант 2*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату.

1. Создайте спецификацию на разработку программного комплекса. Результаты разработки поместите в текстовый документ. Дайте файловому документу имя «Постановка задачи». Этот файл потребуется Вам для выполнения следующих лабораторных практикумов.

2. Составьте описание возможностей разрабатываемого комплекса. Результаты разработки поместите в текстовый документ «Постановка задачи».

3. Составьте описание входных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

4. Составьте описание выходных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

5. Составьте описание ошибок, которые могут возникнуть при работе с программой. Результаты поместите в текстовый документ «Постановка задачи».

6. Составьте контрольный пример для проверки правильности работы программы. Результаты разработки поместите в текстовый документ «Постановка задачи».

*Вариант 3*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех.

1. Создайте спецификацию на разработку программного комплекса. Результаты разработки поместите в текстовый документ. Дайте файловому документу имя «Постановка задачи». Этот файл потребуется Вам для выполнения следующих лабораторных практикумов.

2. Составьте описание возможностей разрабатываемого комплекса. Результаты разработки поместите в текстовый документ «Постановка задачи».

3. Составьте описание входных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

4. Составьте описание выходных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

5. Составьте описание ошибок, которые могут возникнуть при работе с программой. Результаты поместите в текстовый документ «Постановка задачи».

6. Составьте контрольный пример для проверки правильности работы программы. Результаты разработки поместите в текстовый документ «Постановка задачи».

*Вариант 4*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой–участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде-участнице, а также упорядоченный список в соответствии с набранным

количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5.

1. Создайте спецификацию на разработку программного комплекса. Результаты разработки поместите в текстовый документ. Дайте файловому документу имя «Постановка задачи». Этот файл потребуется Вам для выполнения следующих лабораторных практикумов.

2. Составьте описание возможностей разрабатываемого комплекса. Результаты разработки поместите в текстовый документ «Постановка задачи».

3. Составьте описание входных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

4. Составьте описание выходных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

5. Составьте описание ошибок, которые могут возникнуть при работе с программой. Результаты поместите в текстовый документ «Постановка задачи».

6. Составьте контрольный пример для проверки правильности работы программы. Результаты разработки поместите в текстовый документ «Постановка задачи».

*Вариант 5*

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы.

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков.

1. Создайте спецификацию на разработку программного комплекса. Результаты разработки поместите в текстовый документ. Дайте файловому документу имя «Постановка задачи». Этот файл потребуется Вам для выполнения следующих лабораторных практикумов.

2. Составьте описание возможностей разрабатываемого комплекса. Результаты разработки поместите в текстовый документ «Постановка задачи».

3. Составьте описание входных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

4. Составьте описание выходных данных для разрабатываемого комплекса. Результаты поместите в текстовый документ «Постановка задачи».

5. Составьте описание ошибок, которые могут возникнуть при работе с программой. Результаты поместите в текстовый документ «Постановка задачи».

6. Составьте контрольный пример для проверки правильности работы программы. Результаты разработки поместите в текстовый документ «Постановка задачи».

## Лабораторный практикум № 6 Структурный подход к разработке программного обеспечения. Проектирование структуры программы

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 3.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить на практическом примере модели разработки ПО при структурном подходе к проектированию.

***Вводная часть***

*Сущность структурного подхода*

Сущность структурного подхода к разработке информационной системы заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые, в свою очередь, делятся на подфункции, подразделяемые на задачи, и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимоувязаны. При разработке системы "снизу-вверх" от отдельных задач ко всей системе целостность теряется, возникают проблемы при информационной стыковке отдельных компонентов.

Все наиболее распространенные методологии структурного подхода базируются на ряде общих принципов. В качестве двух базовых принципов используются следующие:

- принцип *"разделяй и властвуй"* – принцип решения сложных проблем путем их разбиения на множество меньших независимых задач, легких для понимания и решения;
- принцип *иерархического упорядочивания* – принцип организации составных частей проблемы в иерархические древовидные структуры с добавлением новых деталей на каждом уровне.

Выделение двух базовых принципов не означает, что остальные принципы являются второстепенными, поскольку игнорирование любого из них может привести к непредсказуемым последствиям (в том числе и к провалу всего проекта). Основными принципами проектирования также являются:

- принцип *абстрагирования* – заключается в выделении существенных аспектов системы и отвлечения от несущественных;
- принцип *формализации* – заключается в необходимости строгого, методического подхода к решению проблемы;
- принцип *непротиворечивости* – заключается в обоснованности и согласованности элементов;
- принцип *структурирования данных* – заключается в том, что данные должны быть структурированы и иерархически организованы.

В структурном анализе используются в основном две группы средств, иллюстрирующих функции, выполняемые системой, и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм), среди которых наиболее распространенными являются следующие:

- SADT (Structured Analysis and Design Technique) модели и соответствующие функциональные диаграммы;
- DFD (Data Flow Diagrams) диаграммы потоков данных;
- ERD (Entity-Relationship Diagrams) диаграммы "сущность – связь".

Метод SADT представляет собой совокупность правил и процедур, предназначенных для построения *функциональной модели* объекта какой-либо предметной области. Модель SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

При *моделировании потоков данных* (DFD) модель системы определяется как иерархия диаграмм потоков данных, описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи пользователю.

На стадии проектирования ИС модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектуру ПО, структурные схемы программ и диаграммы экранных форм.

***Практическая часть***

*Функциональная модель учебного комплекса «Деканат»*

В первом лабораторном практикуме поставлена задача:

Разработать программный комплекс по обработке результатов сессии на курсе. Программный комплекс позволяет получать следующие сведения:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;

- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

Необходимо создать функциональную модель разрабатываемого программного комплекса. Данный учебный комплекс получил имя «Деканат».

Результатом применения метода SADT является модель, которая состоит из диаграмм, фрагментов текста и глоссария, имеющих ссылки друг на друга.

*Диаграммы* – главные компоненты модели, на которых все функции и интерфейсы представлены как блоки и дуги, соответственно. Управляющая информация входит в блок сверху. Входная информация, которая подвергается обработке, показана с левой стороны блока, а результаты (выход) показаны с правой стороны.

Одной из наиболее важных особенностей метода SADT является постепенное введение все больших уровней детализации по мере создания диаграмм, отображающих модель.

Построение SADT-модели начинается с представления всей системы в виде простейшего компонента – одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок отражает систему как единое целое, имя, указанное в блоке, является общим.

Функциональная диаграмма программного комплекса «Деканат» представлена на рисунке 11.

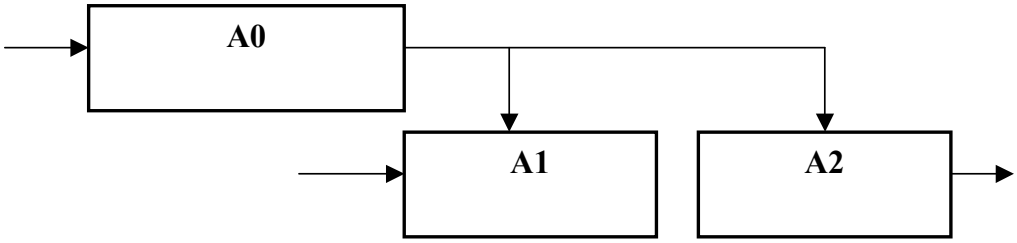



Рисунок 11. Функциональная диаграмма программного комплекса «Деканат»

Модель SADT представляет собой *серию диаграмм с сопроводительной информацией*, в которой расшифровывается назначение блоков диаграммы.

Сопроводительная информация для функциональной диаграммы программного комплекса «Деканат»:

A0 – управляющая программа. Эта программа выполняет функции диспетчера. Организует работу двух подчиненных ей модулей.

A1 – модуль ввода исходной информации. В модуле осуществляется проверка входной информации на допустимые значения. Информация записывается на диск.

A2 – модуль обработки исходной информации с целью получения запрашиваемых величин. Подлежит уточнению.

*Модель потока данных*

Диаграммы потоков данных (DFD) являются основным средством моделирования функциональных требований к проектируемой системе. С их помощью эти требования представляются в виде иерархии функциональных компонентов (процессов), связанных потоками данных. Главная цель такого представления – продемонстрировать, как каждый процесс преобразует свои входные данные в выходные, а также выявить отношения между этими процессами.

Диаграммы верхних уровней иерархии (контекстные диаграммы) определяют основные процессы или подсистемы с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая многоуровневую иерархию диаграмм, до тех пор, пока не будет достигнут такой уровень декомпозиции, на котором процессы становятся элементарными и детализировать их далее невозможно.

Диаграмма потоков данных программного комплекса «Деканат» представлена на рисунке 12.

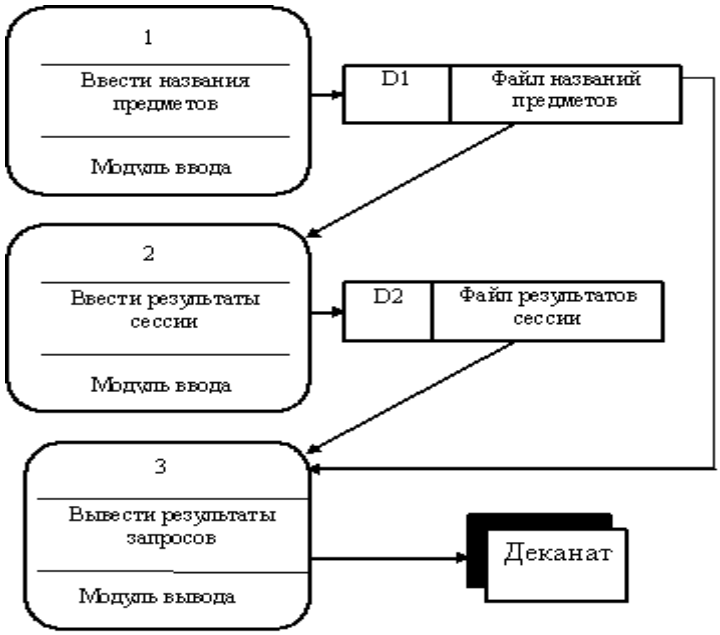



Рисунок 12. Диаграмма потоков данных программного комплекса «Деканат»

Диаграмма потоков данных должна быть снабжена миниспецификацией.

*Миниспецификация* (описание логики процесса) должна формулировать его основные функции таким образом, чтобы в дальнейшем специалист, выполняющий реализацию проекта, смог выполнить их или разработать соответствующую программу.

Миниспецификация является конечной вершиной иерархии ДПД. Решение о завершении детализации процесса и использовании миниспецификации принимается аналитиком исходя из следующих критериев:

- наличия у процесса относительно небольшого количества входных и выходных потоков данных (2–3 потока);
- возможности описания преобразования данных процессом в виде последовательного алгоритма;
- выполнения процессом единственной логической функции преобразования входной информации в выходную;
- возможности описания логики процесса при помощи миниспецификации небольшого объема (не более 20–30 строк).

*Миниспецификация для диаграммы потоков данных программного комплекса «Деканат»*

D1. Файл названий предметов:

| | Данные | Условное обозначение | Тип |
|---|---|---|---|
| 1 | Количество предметов, сданных в сессию | kol | Целый |
| 2 | Названия предметов | Pred (массив) | Строковый |

Файл может быть текстовым.

D2. Файл результатов сессии.

| | Данные | Условное обозначение | Тип |
|---|---|---|---|
| 1 | Номер группы | N_G | Целый |
| 2 | Фамилия | F | Строковый |
| 3 | Имя | Name1 | Строковый |
| 4 | Отчество | Name2 | Строковый |
| 5 | Оценки | Rez (массив) | Целый (1, 2, 3, 4, 5) |

В связи с тем, что по данным файла создаются запросы, следует организовать данные в файл прямого доступа. В качестве компоненты файла использовать структуру «Запись».

***Самостоятельная работа***

*Вариант 1*

Разработать программный комплекс «Расписание», который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота).

1. Составьте функциональную модель программного комплекса «Расписание». Рисунок выполните в текстовом редакторе OpenOffice.org Write.
2. Составьте сопроводительную информацию к функциональной модели. Результат оформите в текстовом редакторе OpenOffice.org Write.
3. Составьте модель потока данных программного комплекса «Расписание». Рисунок выполните в текстовом редакторе OpenOffice.org Write.
4. Составьте миниспецификацию для модели потока данных программного комплекса «Расписание». Результат оформите в текстовом редакторе OpenOffice.org Write.

*Вариант 2*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату.

1. Составьте функциональную модель программного комплекса «Отдел кадров». Рисунок выполните в текстовом редакторе OpenOffice.org Write.
2. Составьте сопроводительную информацию к функциональной модели. Результат оформите в текстовом редакторе OpenOffice.org Write.
3. Составьте модель потока данных программного комплекса «Отдел кадров». Рисунок выполните в текстовом редакторе OpenOffice.org Write.
4. Составьте миниспецификацию для модели потока данных программного комплекса «Отдел кадров». Результат оформите в текстовом редакторе OpenOffice.org Write.

*Вариант 3*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех.

1. Составьте функциональную модель программного комплекса по обслуживанию риэлтовской компании. Рисунок выполните в текстовом редакторе OpenOffice.org Write.
2. Составьте сопроводительную информацию к функциональной модели. Результат оформите в текстовом редакторе OpenOffice.org Write.
3. Составьте модель потока данных программного комплекса по обслуживанию риэлтовской компании. Рисунок выполните в текстовом редакторе OpenOffice.org Write.
4. Составьте миниспецификацию для модели потока данных программного комплекса по обслуживанию риэлтовской компании. Результат оформите в текстовом редакторе OpenOffice.org Write.

*Вариант 4*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой–участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде–участнице, а также упорядоченный список в соответствии с набранным

количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5.

1. Составьте функциональную модель программного комплекса для подведения итогов Олимпийских игр. Рисунок выполните в текстовом редакторе OpenOffice.org Write.

2. Составьте сопроводительную информацию к функциональной модели. Результат оформите в текстовом редакторе OpenOffice.org Write.

3. Составьте модель потока данных программного комплекса для подведения итогов Олимпийских игр. Рисунок выполните в текстовом редакторе OpenOffice.org Write.

4. Составьте миниспецификацию для модели потока данных программного комплекса для подведения итогов Олимпийских игр. Результат оформите в текстовом редакторе OpenOffice.org Write.

*Вариант 5*

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков.

1. Составьте функциональную модель программного комплекса для обработки информации о подписных изданиях. Рисунок выполните в текстовом редакторе OpenOffice.org Write.

2. Составьте сопроводительную информацию к функциональной модели. Результат оформите в текстовом редакторе OpenOffice.org Write.

3. Составьте модель потока данных программного комплекса для обработки информации о подписных изданиях. Рисунок выполните в текстовом редакторе OpenOffice.org Write.

4. Составьте миниспецификацию для модели потока данных программного комплекса для обработки информации о подписных изданиях. Результат оформите в текстовом редакторе OpenOffice.org Write.

## Лабораторный практикум № 7 Структурный подход к разработке программного обеспечения. Программная реализация при нисходящей разработке

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 3.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить на практическом примере процесс кодирования при нисходящей разработке ПО.

***Вводная часть***

*Стиль программирования*

Важнейшая технологическая задача, возникающая в процессе программирования, – соответствие единому стилю программирования. Под стилем программирования обычно понимают набор приемов или методов программирования, которые используют опытные программисты, чтобы получить правильные, эффективные, удобные для применения и легкочитаемые программы. Правила хорошего стиля – результат соглашения между опытными программистами. Обычно принципы хорошего стиля программирования являются результатом здравого смысла, исходящего из опыта. Код должен быть прост и понятен, т.е. обладать следующими свойствами:

- очевидная логика;
- естественные выражения;

- использование соглашений, принятых в языке разработки;
- осмысленные имена;
- аккуратное форматирование;
- развернутые комментарии;
- отсутствие хитрых трюков и необычных конструкций.

Правило стандартизации стиля заключается в том, что если существует более одного способа сделать что–либо и выбор произвольный, то следует остановиться на одном способе и всегда его придерживаться. Особое значение имеет единый стиль программирования в процессе работы над программным текстом в коллективных разработках. В большинстве крупных проектов существуют внутренние документы, определяющие стиль программирования команды разработчиков.

Хорошего программиста отличает способность писать удобочитаемые программы. Небольшие усилия, которые необходимо приложить для того, чтобы сделать программу удобочитаемой, обходятся дешевле, чем издержки по пересмотру, обнаружению ошибок или переделке плохо написанной программы.

*Комментарии*

**Программа должна содержать комментарии. Их часто опускают с целью экономии времени. Программы с пояснительными комментариями значительно легче отлаживать, так как они содержат дополнительную информацию. Существует три вида комментариев: вводные, оглавления, пояснительные.**

*Вводные комментарии*

Каждая программа, подпрограмма должна начинаться с комментариев, поясняющих, что она делает.

*Минимальная информация*

1. Назначение программы.
2. Указания по вызову программы и ее использованию.
3. Список и назначение основных переменных или массивов.
4. Указания по вводу/выводу. Список всех файлов.
5. Список используемых подпрограмм.
6. Название применяемых математических методов, а также ссылки на литературные источники.
7. Сведения о времени выполнения программы
8. Требуемый объем памяти.
9. Специальные указания оператору.
10. Сведения об авторе.
11. Дата создания.

*Пояснительные комментарии*

Пояснениями нужно сопровождать те части программы, которые трудно понять без комментариев. Комментировать следует каждый логически выделенный модуль программы.

Существенно содержание комментариев. Не нужно переводить с английского операторы языка – нужно указывать цель оператора.

*Расположение комментариев*

Комментарии, которые перемежаются с текстом программы, легче читать, когда они выделены пустыми строками. Комментарии можно заключить в прямоугольник из специальных символов. Можно заключить в прямоугольник группу команд, к которым относится комментарий. Располагать комментарии нужно так, чтобы улучшить наглядность программы, а не ухудшить.

*Пропуск строк*

Как в естественном языке мы пользуемся пропуском строк для отделения параграфов, так и в программе можно разделять ее отдельные фрагменты. Пропуском одной строки можно отделять каждую группу логически связанных операторов, пропуском двух строк – основные логические фрагменты программы.

*Пробелы*

В языках программирования пробелы довольно часто ставятся произвольно. Действительно, в изъятии пробелов из программы не больше смысла, чем в том, чтобы их убрать из текста.

Широкое использование пробелов существенно облегчает чтение программы. Пробелы следует ставить между элементами списка данных, а также до и после операций +, –, =. Иногда желательно отделять пробелами операции (*,/). Пробелы можно также использовать для указания приоритета операций. Например, запись вида

```
1 + A*B
```

предпочтительнее, чем вводящее в заблуждение выражение

```
1 + A * B
```

*Выбор имен переменных*

Имена должны давать представления о том, что за величины за ними скрываются. Например, лучше использовать

```
Const Price = 508.00;
```

вместо

```
Const X = 508.00;
```

Нужно избегать схожих по виду имен:

phone и fone

AX10 и AXI0

Различие имен должно быть всегда явно ощутимым.

В качестве имен переменных должны употребляться термины, используемые в данной области. Хорошим тоном считается использование разделителя. В Паскале это знак подчеркивания:

P_List, Name_One, Volume_Bottom

Такие имена облегчают чтение и уменьшают вероятность неправильной интерпретации.

Это относится и к именам файлов. Имена файлов должны отличаться уже первыми восемью символами. Некоторые устаревшие, но тем не менее широко используемые операционные системы (MS DOS) накладывают ограничения на длину имени файла. Все файлы должны иметь различные имена, даже если они находятся в разных каталогах.

*Размещение операторов*

Лучше размещать операторы по одному в строке. В таком случае легче локализовать ошибки.

Читаемость улучшается, если для записи операторов используется отступ. Операторы, связанные между собой, записываются с одинаковым отступом.

Отступы, не оказывая влияния на логику программы, существенно улучшают ее читаемость. Программа должна быть приятна для глаза.

*Защитное программирование*

*Защитное программирование* – это такой стиль написания программ, при котором появляющиеся ошибки легко обнаруживаются и идентифицируются программистом. Существует три основных принципа защитного программирования.

- Общее недоверие. Для каждого модуля входные данные должны тщательно анализироваться в предположении, что они могут быть ошибочными.
- Немедленное обнаружение. Каждая ошибка должна быть выявлена как можно раньше, это упрощает установление ее причины.
- Изолирование ошибки. Ошибки в одном модуле должны быть изолированы, чтобы не допустить их влияние на другие модули.

Рекомендации:

- Делайте проверку области значений переменных.
- Выполняйте контроль правдоподобности значений переменных, которые не должны превышать некоторых констант или значений других переменных.
- Проверяйте длину элементов информации.
- Проверяйте коды возврата функций.

*Удобство работы с программой*

Большое значение имеют удобный ввод и красивый вывод.

Ввод должен быть снабжен подсказкой и защитой от неверных значений.

Вывод должен быть удобным для восприятия информации и не слишком большим. Большие тексты редко дочитываются до конца.

*Кодирование и тестирование сверху вниз*

*Нисходящая разработка* – это подход к разработке программного комплекса, при котором он разбивается на программные модули (программы), образующие многоуровневую структуру (не путать с понятием «модуль», которое используется для определения синтаксической конструкции языка). Каждый программный модуль представляет собой короткую программу, решающую отдельную задачу (подзадачу). Главная программа должна быть короткой и вызывать модули и подпрограммы, которые можно моделировать, создавая подыгрывающие подпрограммы («заглушки»). *Подыгрывающая программа* (*заглушка*) – очень короткая последовательность команд, которая используется как замена, пока не будет создана фактическая программа.

В любой момент разработки программного комплекса имеется его действующий вариант. Тестирование и отладка отдельных программных модулей и программного комплекса в целом ведется по ходу его проектирования.

***Практическая часть***

*Кодирование программного комплекса «Деканат»*

При разработке программного комплекса «Деканат» получена функциональная диаграмма первого уровня детализации. В полученной диаграмме подсистемы «Ввод» и «Запрос» не уточнены. На этом этапе имеется возможность перейти к кодированию.

Кодирование программного комплекса начинается с управляющего программного модуля. Для его тестирования и отладки необходимо иметь программные модули второго уровня «Ввод» и «Запрос», но, так как они еще не спроектированы и не закодированы, вместо них используются имитаторы этих программных модулей – «заглушки». Так как назначение «заглушек» только в том, чтобы программный модуль верхнего уровня был выполнен, они могут быть достаточно простыми. В приведенном примере имеется возможность определить структуры данных.

Ознакомьтесь с предлагаемой программой. Введите код и выполните программу в среде PascalABC.

Программа на языке PascalABC:

```
{-------------------------------------------------------------------------------------------------------------}
{-------------Подсистема комплекса "Обработка результатов сессии"-------------------------}
{-------------------------Учебная программа для демонстрации-----------------------------------}
{-------------------------структурного подхода к разработке ПО-----------------------------------}
{---------------Разработчик – НИИ компьютерного обучения ---2015г.--------------------}
{---------------------------------------UchKomplex.pas-------------------------------------------------}
Program UchKompleks;
Uses crt;
Const N_U = 20;
                                        {Максимальное количество обучающихся на курсе – 20}
                                        {Число, достаточное для учебной задачи}
Type TStr = String[20];
     TPred = array [1..6] Of TStr;
                                        {Тип – массив для описания наименований предметов}
     Fio = Record                       {Структура для определения имени}
             F, Name1, Name2: TStr;
           End;
     Sess = Record                              {Тип компоненты файла результатов сессии}
             N_G: byte;                         {Номер группы}
```

```
            Name: Fio;                  {Имя обучающегося}
            Rez: Array [1..6] Of Byte;  {Оценки}
          End;
Var
  Pred: TPred;              {Наименования предметов}
  FName: TStr;              {Имя Файла}
  Kol_Pr: Integer;          {Количество предметов}
  n_Zap: Integer;           {Количество обучающихся}
  Reg: Integer;             {Для организации меню}
Begin
  Repeat
                  {Цикл для организации обращения к блокам системы}
    Writeln('Выберите режим работы');
    Writeln(1,' ':10,'Ввод');
    Writeln(2,' ':10,'Запрос');
    Writeln(3,' ':10,'Конец работы');
    Write(' ':10,'Режим?'); Readln(Reg);
    Case reg Of
      1: Begin Writeln('Блок ввода'); Delay(2000); End;          {Заглушка 1}
      2: Begin Writeln('Блок запросов'); Delay(2000); End;       {Заглушка 2}
      3: Begin Writeln('Конец работы'); Delay(2000); End;
    End;
  Until Reg = 3;
End.
```

***Самостоятельная работа***

*Вариант 1*

Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема: позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота). Результат запишите в рабочую директорию.

На основании составленной на прошлом лабораторном практикуме спецификации к разработке программного комплекса, функциональной диаграмме и диаграмме потока данных составьте описание структур данных и программу на PascalABC, реализующую обращение к подчиненном модулям.

В соответствии с принятым стилем в программах необходимо:

- использовать вводные и пояснительные комментарии;
- делать пропуск строк для разделения групп логически связанных операторов;
- делать пробелы для улучшения читаемости программы;
- для переменных задавать имена «со смыслом»;
- при размещении делать одинаковые отступы в строке для связанных операторов.

1. Создайте код программы.

2. Введите текст программы в текстовый редактор системы PascalABC. Выполните программу.

*Вариант 2*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату. Результат запишите в рабочую директорию.

На основании составленной на прошлом лабораторном практикуме спецификации к разработке программного комплекса, функциональной диаграмме и диаграмме потока данных составьте описание структур данных и программу на PascalABC, реализующую обращение к подчиненном модулям.

В соответствии с принятым стилем в программах необходимо:

- использовать вводные и пояснительные комментарии;
- делать пропуск строк для разделения групп логически связанных операторов;
- делать пробелы для улучшения читаемости программы;
- для переменных задавать имена «со смыслом»;
- при размещении делать одинаковые отступы в строке для связанных операторов.

1. Создайте код программы.

2. Введите текст программы в текстовый редактор системы PascalABC. Выполните программу.

*Вариант 3*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех. Результат запишите в рабочую директорию.

На основании составленной на прошлом лабораторном практикуме спецификации к разработке программного комплекса, функциональной диаграмме и диаграмме потока данных составьте описание структур данных и программу на PascalABC, реализующую обращение к подчиненном модулям.

В соответствии с принятым стилем в программах необходимо:

- использовать вводные и пояснительные комментарии;
- делать пропуск строк для разделения групп логически связанных операторов;
- делать пробелы для улучшения читаемости программы;
- для переменных задавать имена «со смыслом»;
- при размещении делать одинаковые отступы в строке для связанных операторов.

1. Создайте код программы.

2. Введите текст программы в текстовый редактор системы PascalABC. Выполните программу.

*Вариант 4*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой–участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде–участнице, а также упорядоченный список в соответствии с набранным количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5. Результат запишите в рабочую директорию.

На основании составленной на прошлом лабораторном практикуме спецификации к разработке программного комплекса, функциональной диаграмме и диаграмме потока данных составьте описание структур данных и программу на PascalABC, реализующую обращение к подчиненном модулям.

В соответствии с принятым стилем в программах необходимо:

- использовать вводные и пояснительные комментарии;
- делать пропуск строк для разделения групп логически связанных операторов;
- делать пробелы для улучшения читаемости программы;
- для переменных задавать имена «со смыслом»;
- при размещении делать одинаковые отступы в строке для связанных операторов.

1. Создайте код программы.

2. Введите текст программы в текстовый редактор системы PascalABC. Выполните программу.

*Вариант 5*

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков. Результат запишите в рабочую директорию.

На основании составленной на прошлом лабораторном практикуме спецификации к разработке программного комплекса, функциональной диаграмме и диаграмме потока данных составьте описание структур данных и программу на PascalABC, реализующую обращение к подчиненном модулям.

В соответствии с принятым стилем в программах необходимо:

- использовать вводные и пояснительные комментарии;
- делать пропуск строк для разделения групп логически связанных операторов;
- делать пробелы для улучшения читаемости программы;
- для переменных задавать имена «со смыслом»;
- при размещении делать одинаковые отступы в строке для связанных операторов.

1. Создайте код программы.

2. Введите текст программы в текстовый редактор системы PascalABC. Выполните программу.

## Лабораторный практикум № 8 Структурный подход к разработке программного обеспечения. Модульный подход к проектированию и программированию программного обеспечения

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 3.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** научиться применять модульный подход к проектированию и программированию ПО.

***Вводная часть***

*Свойства программных модулей*

При использовании технологии нисходящего структурного программирования появляется возможность включать в разрабатываемые программные комплексы, созданные ранее, программные модули.

Использование небольших программных модулей имеет преимущества. С такими модулями удобнее работать, они позволяют разрабатывать программные комплексы, которые легче модифицировать. Небольшие модули легче и эффективнее тестируются.

Свойства программных модулей:

1. Программный модуль должен иметь один вход и один выход.

2. Программный модуль должен решать самостоятельную задачу по принципу один программный модуль – одна функция. Например, в разрабатываемом комплексе «Деканат» имеются следующие модули: «Ввод», «Запрос 1», «Запрос 2», «Запрос 3», «Запрос 4», «Модуль ввода числа целого типа».

3. Работа программного модуля не должна зависеть:

– от входных данных;

– от того, какому программному модулю предназначены его выходные данные:

– от предыстории вызовов программного модуля.

4. Программный модуль должен возвращать управление тому программному модулю, который его вызвал.

5. Программный модуль может вызывать другой программный модуль.

6. Размер программного модуля желательно ограничить одной – двумя страницами исходного текста.

Программный модуль должен иметь спецификацию. Содержание спецификации модуля:

1. Назначение модуля.
2. Название модуля.
3. Основные ограничения при применении.
4. Описание и назначение основных функций.
5. Математическое описание.
6. Схемы алгоритмов.
7. Перечень переменных по формам (рисунок 13).

Входные данные

| Название переменной в описании задачи | Наименование переменной | Тип переменной | Форма взаимодействия, с основной программой | Ограничения | Назначение переменной |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| ... | ... | ... | ... | ... | |

Выходные данные

| Название переменной в описании задачи | Наименование переменной | Тип переменной | Форма взаимодействия с основной программой | Назначение переменной |
|---|---|---|---|---|
| | | | | |
| | | | | |
| ... | ... | ... | ... | ... |

Внутренние переменные

| Название переменной в описании задачи | Наименование переменной | Тип переменной | Назначение переменной |
|---|---|---|---|
| | | | |
| | | | |
| ... | ... | ... | ... |

Рисунок 13. Спецификация модуля

8. Способы подключения модуля к основной программе.

*Интерфейс модуля* – это те данные, с помощью которых модуль взаимодействует с другими программами (входные и выходные данные, файлы для передачи информации, глобальные переменные).

Если работа над программным продуктом проводится в коллективе разработчиков, то можно распределить обязанности программистов таким образом, что у каждого модуля будет свой разработчик. Главное, чтобы была согласованность во входных и выходных интерфейсах.

*Подключение модулей в программу*

Отдельные программные модули могут оформляться в виде модулей (Unit) или в виде процедур и функций. Существует несколько способов подключения процедур в программу на языке PascalABC.

- Исходный текст процедуры может подключаться непосредственно в текст разрабатываемой программы с помощью редактора, встроенного в систему программирования PascalABC или другого текстового редактора. Использование редактора позволяет внести изменения в текст процедуры.
- Текст одной или нескольких процедур оформляется в виде библиотечного программного модуля, файл которого имеет расширение PAS, и после указания его имени в разделе USES программа получает доступ к процедурам из этого модуля.

После отладки управляющего программного модуля «заглушки» программных модулей второго уровня кодируются и отлаживаются одновременно с управляющей программой. При этом программные модули третьего уровня заменены «заглушками».

***Практическая часть***

*Подключение модулей в учебном комплексе «Деканат»*

В основной программе подключаются два модуля: Vvod –модуль, который будет осуществлять ввод исходной информации; Zapros – модуль, который будет обслуживать запросы программного комплекса.

Заменяем заглушки первого уровня обращениями к подпрограммам, расположенным в модулях. Vvodi – подпрограмма ввода. Zaprosi – подпрограмма запросов. В подпрограмме Vvodi вводится имя файла. В подпрограмме Zaprosi – имя файла выводится. Таким образом, проверяется правильность передачи информации из одного модуля в другой.

В программе дополнения выделены жирным курсивом.

*Основная программа комплекса:*

```
{-------------------------------------------------------------------------------------------}
{----------Подсистема комплекса «Обработка результатов сессии»-------------}
{-----------------------------------------Этап 1-----------------------------------------}
Program UchKompleks;
Uses crt, Vvod, Zapros; {Подключаем модули к основной программе}
Const N_U = 20;
    {Максимальное количество обучающихся на курсе – 20}
    {Число, достаточное для учебной задачи}
Type    TStr = String[20];
        TPred = array [1..6] Of TStr;
                                {Тип – массив для описания наименований предметов}
        Fio = Record                    {Структура для определения имени}
                F, Name1, Name2: TStr;
        End;
        Sess = Record                           {Тип компоненты файла результатов сессии}
                N_G: byte;                      {Номер группы}
                Name: Fio;                      {Имя обучающегося}
                Rez: Array [1..6] Of Byte; {Оценки}
        End;
Var
        Pred: TPred;            {Наименования предметов}
        FName: TStr;            {Имя Файла}
        Kol_Pr: Integer;  {Количество предметов}
        n_Zap: Integer;   {Количество обучающихся}
        Reg: Integer;           {Для организации меню}
Begin
 Repeat
  Writeln('Выберите режим работы);
  Writeln(1,' ':10,'Ввод');
  Writeln(2,' ':10,'Вывод');
  Writeln(3,' ':10,'Конец работы');
  Write(' ':10,'Режим'); Readln(Reg);
  Case reg Of
   1: Begin vvodi(FName); Delay(4000); End;         {Обращение к подпр. из модуля Vvod}
   2: Begin Zaprosi(FName); Delay(4000); End; { Обращение к подпр. из модуля Zapros }
   3: Begin Writeln('Конец работы'); Delay(2000); End;
 End;
 Until Reg = 3;
End.
```

В модуль ввода копируем описание типов данных. Это необязательное действие. Типы данных потребуются при дальнейшей детализации программы.

В модуле ввода вводится имя файла FName, которое должно быть получено при выводе в модуле вывода.

*Модуль ввода*

```
Unit Vvod;
{--------------------------------------Vvod-------------------------------------}
{-------------------------Ввод результатов сессии-------------------------}
Const N_U = 20;
Type     TStr = String[20];
           TPred = array [1..6] Of TStr;{Тип данных для предметов}
           Fio = Record
           F, Name1, Name2: TStr;
         End;
           Sess = Record
           N_G: byte;
           Name: Fio;
           Rez: Array [1..6] Of Byte;
          End;
Procedure Vvodi (Var FName:TStr);
Begin
  Writeln('Имя текстового файла - predmet.dat');
  Writeln('Имя файла результатов сессии?'); Readln(Fname);
  Writeln(' Имя файла результатов сессии - ',Fname);
End;
End.
```

Так же, как и в модуль ввода, в модуль запросов включаем описание типов данных.

*Модуль запросов*

```
Unit Zapros;
Uses Crt;
{-----------Проверка передачи информации в модуль запросов ----------}
Const N_U = 20;
Type     TStr = String[20];
           TPred = array [1..6] Of TStr;{ Тип данных для предметов}
           Fio = Record
           F, Name1, Name2: TStr;
         End;
           Sess = Record
           N_G: byte;
           Name: Fio;
           Rez: Array [1..6] Of Byte;
          End;
Procedure Zaprosi(FName: String);
Begin
           Clrscr;
           Writeln('Имя текстового файла - predmet.dat');
           Writeln('Имя файла результатов сессии - ',Fname);
End;
End.
```

*Продолжение структурной декомпозиции учебной задачи*

На стадии проектирования информационной системы модели расширяются, уточняются и дополняются диаграммами, отражающими структуру программного обеспечения: архитектуру ПО, структурные схемы программ и диаграммы экранных форм.

Одной из наиболее важных особенностей метода SADT является постепенное введение все больших уровней детализации по мере создания диаграмм, отображающих модель.

Построение SADT-модели начинается с представления всей системы в виде простейшего компонента – одного блока и дуг, изображающих интерфейсы с функциями вне системы. Поскольку единственный блок отражает систему как единое целое, имя, указанное в блоке, является общим.

Затем блок, который представляет систему в качестве единого модуля, детализируется на другой диаграмме с помощью нескольких блоков, соединенных интерфейсными дугами. Эти блоки определяют основные подфункции исходной функции. Данная декомпозиция выявляет полный набор подфункций, каждая из которых показана как блок, границы которого определены интерфейсными дугами. Каждая из этих подфункций может быть декомпозирована подобным образом в целях большей детализации. Каждая детальная диаграмма является декомпозицией блока из диаграммы предыдущего уровня.

В первом лабораторном практикуме проведена частичная декомпозиция поставленной задачи разработки программного комплекса «Деканат». Декомпозиция должна быть продолжена. Уточнению должен быть подвержен модуль «Запрос», так как этот модуль выполняет комплексную задачу.

На рисунке 14 представлена функциональная диаграмма модуля «Запрос» программного комплекса «Деканат».

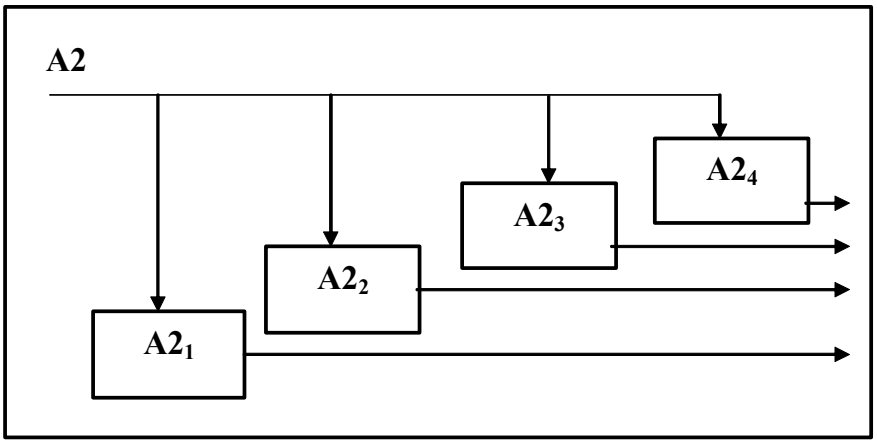



Рисунок14. Функциональная диаграмма модуля «Запрос» программного комплекса «Деканат»

Спецификация диаграммы:

$A2_1$ – выполнение 1-го запроса;

$A2_2$ – выполнение 2-го запроса;

$A2_3$ – выполнение 3-го запроса;

$A2_4$ – выполнение 4-го запроса.

Уточнению подлежит и диаграмма потоков данных. Диаграммы верхних уровней иерархии определяют основные процессы или подсистемы с внешними входами и выходами. Они детализируются при помощи диаграмм нижнего уровня. Такая декомпозиция продолжается, создавая иерархию диаграмм, до тех пор, пока не будет достигнут уровень декомпозиции, на котором процессы становятся элементарными и детализировать их далее невозможно.

На рисунке 15 представлена диаграмма потоков данных модуля «Запрос» программного комплекса «Деканат».

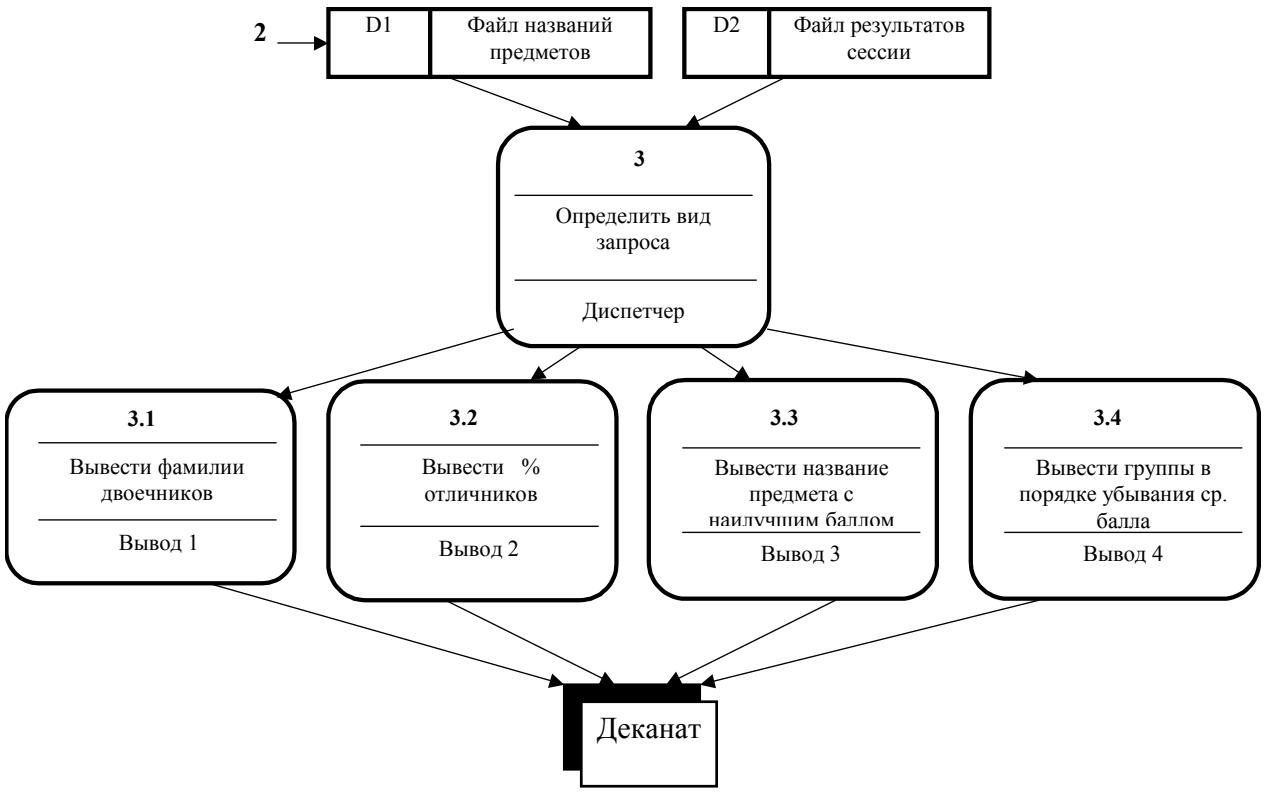



Рисунок 15. Диаграмма потоков данных модуля «Запрос» программного комплекса «Деканат»

Поскольку учебный пример не имеет такой сложной структуры, как промышленная разработка, на данном этапе можно перейти к построению алгоритмов. Схемы алгоритмов модулей программного комплекса «Деканат» представлены на рисунках 16–22.

В языках программирования, ориентированных на технологию нисходящего структурного программирования, одним из средств реализации модульной структуры являются процедуры и функции.

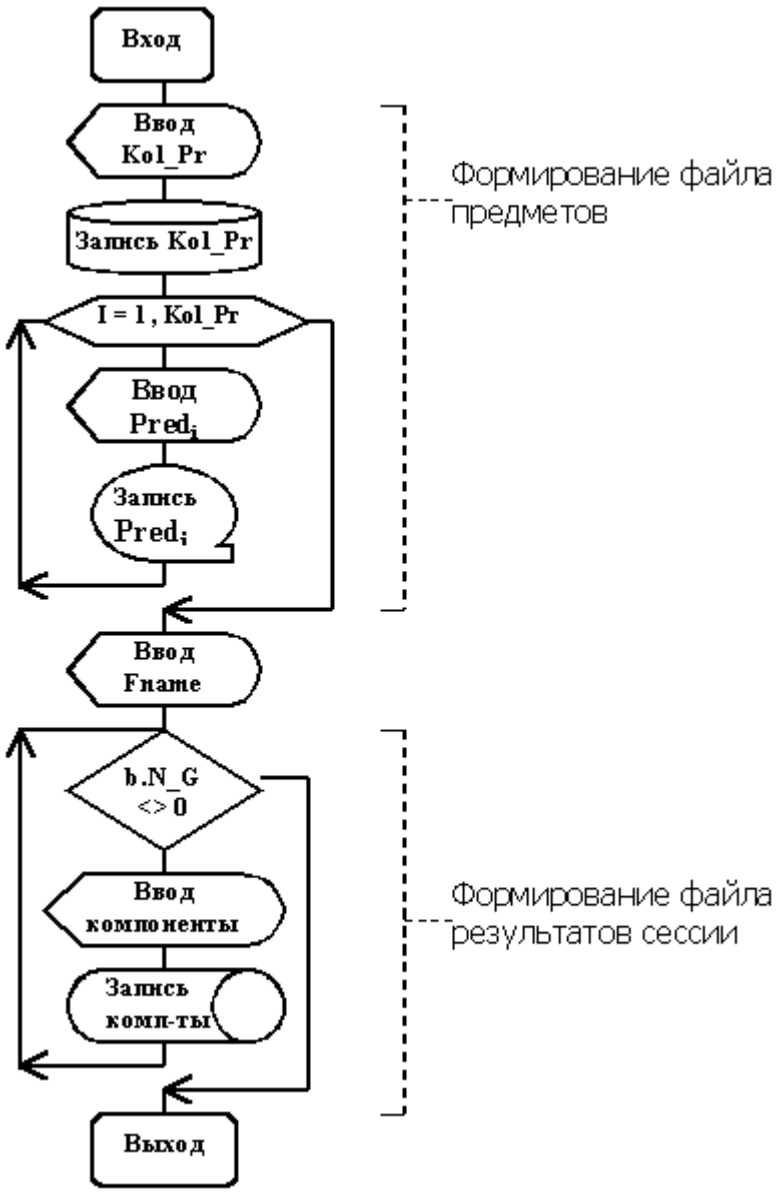



Рисунок 16. Схема модуля «Ввод»

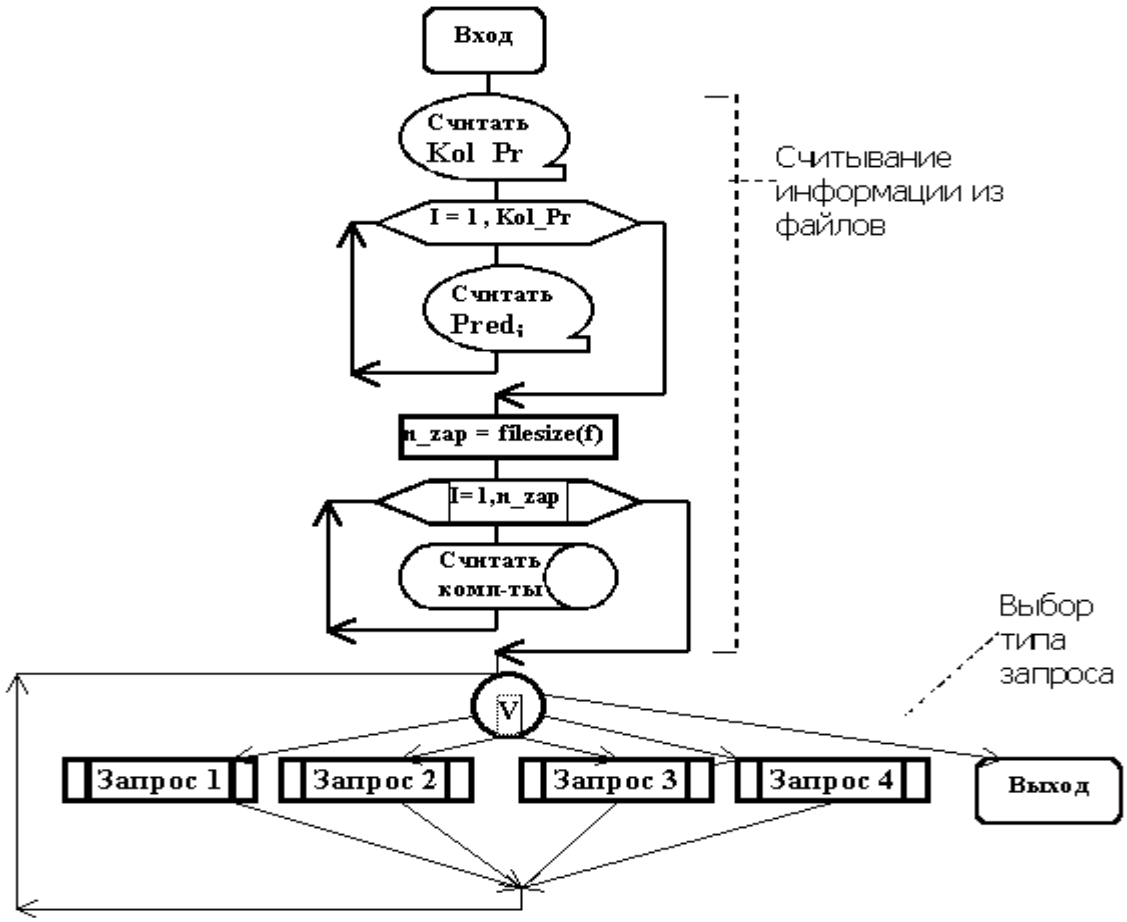



Рисунок 17. Схема модуля «Запрос»

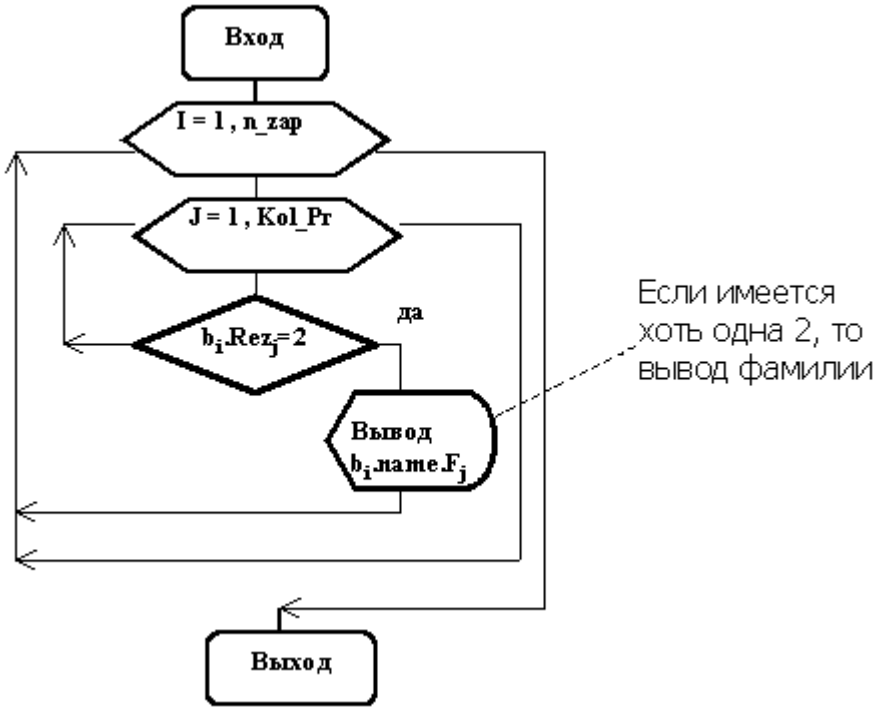



Рисунок 18. Схема модуля «Запрос 1» – «Вывод фамилий двоечников»

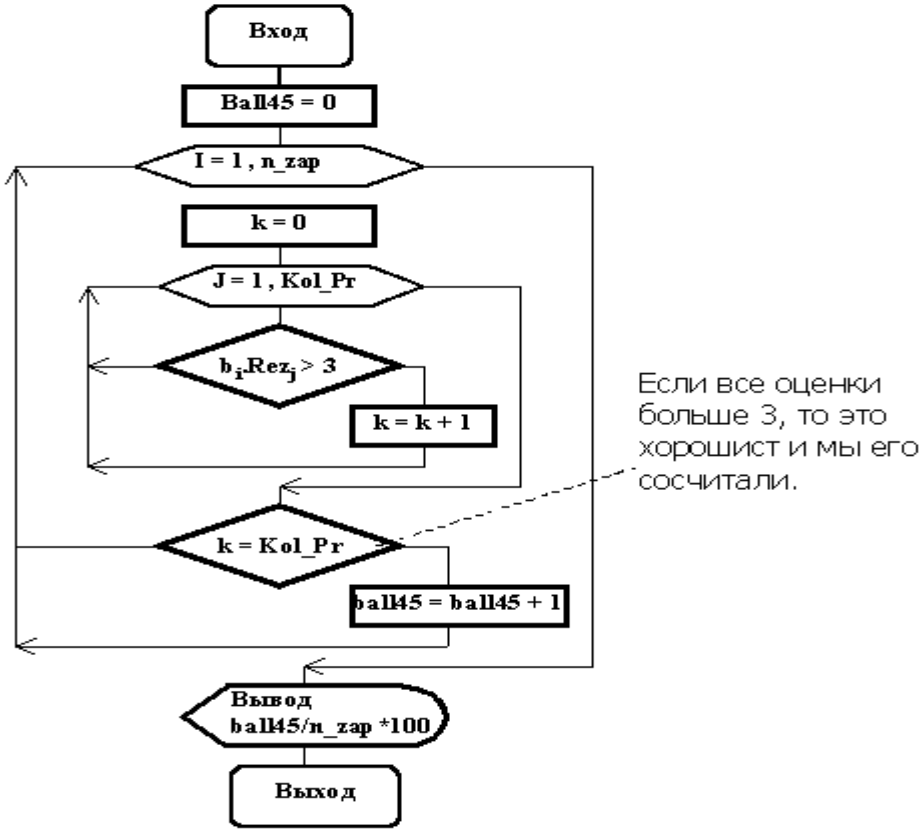



Рисунок 19. Схема модуля «Запрос 2» – «Процент отличников и хорошистов»

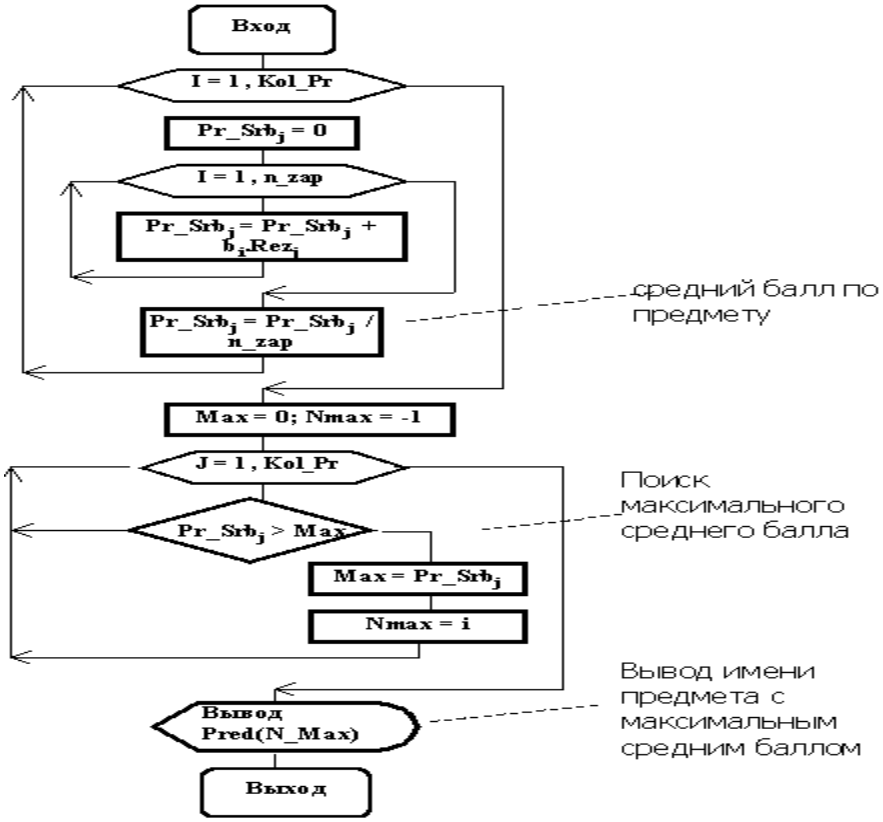



Рисунок 20. Схема модуля «Запрос 3» – «Название предмета с наилучшим средним баллом»

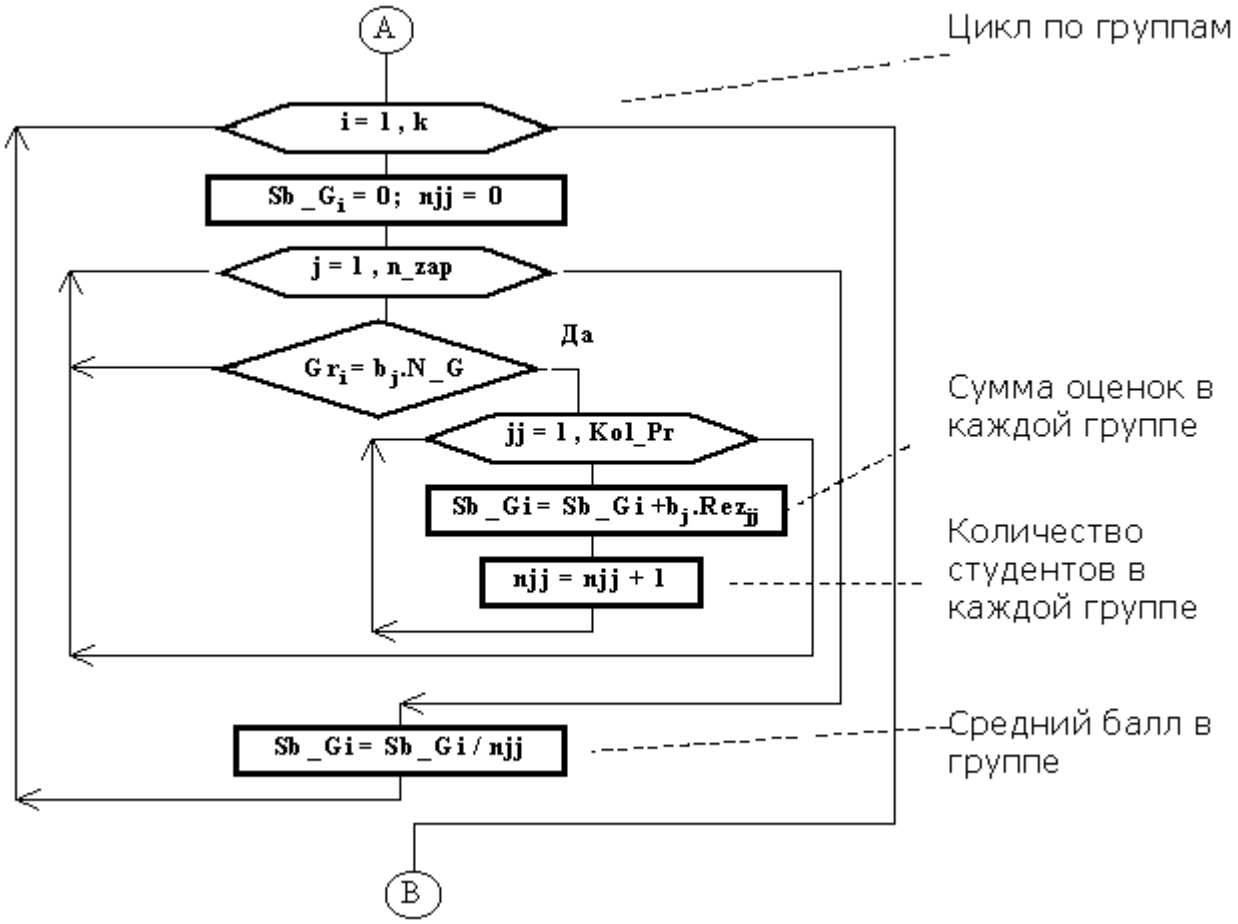



Рисунок 21. Схема блока «Расчет среднего балла»

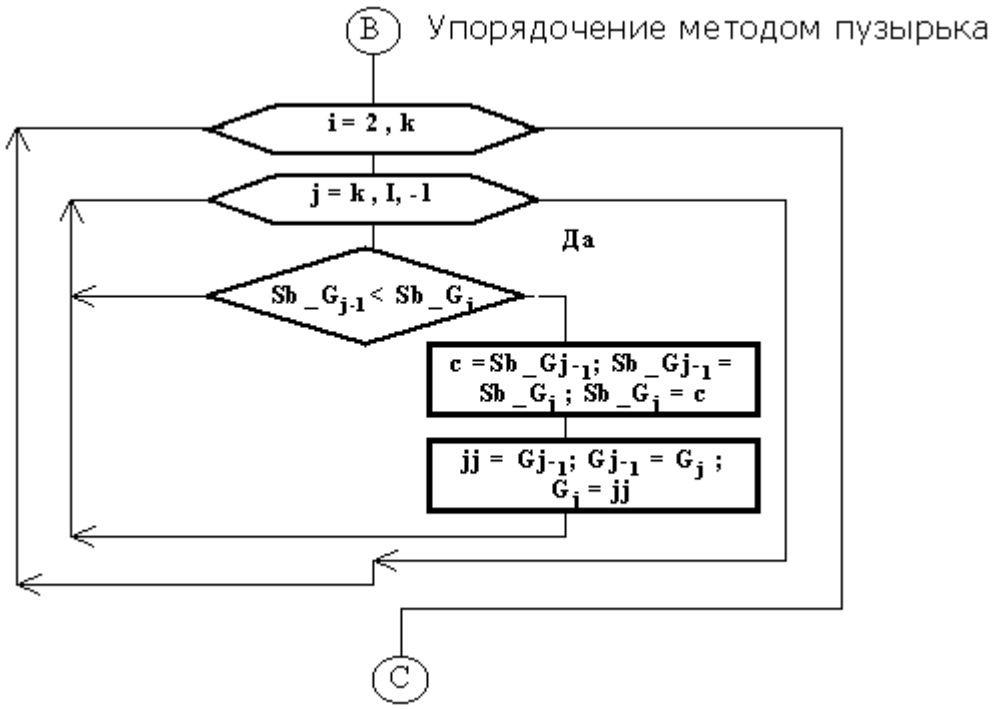



Рисунок 22. Схема блока «Упорядочение по убыванию среднего балла»

***Самостоятельная работа***

*Вариант 1*

Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота).

1. Скорректируйте программу, созданную на предыдущем лабораторном практикуме. Замените заглушки (операторы типа: Writeln('Блок ввода'); , Writeln('Блок запросов');) на обращение к модулям ввода и вывода.
2. Создайте модуль ввода.
3. Создайте модуль вывода.
4. Выполните задачу и осуществите проверку правильности передачи информации из модуля ввода в модуль вывода. Сохраните программу.
5. Проведите второй этап декомпозиции Вашей задачи. Уточните функциональную модель разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.
6. Проведите второй этап декомпозиции Вашей задачи. Уточните модель потока данных разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.

*Вариант 2*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату.

1. Скорректируйте программу, созданную на предыдущем лабораторном практикме. Замените заглушки (операторы типа: Writeln('Блок ввода'); , Writeln('Блок запросов');) на обращение к модулям ввода и вывода.
2. Создайте модуль ввода.
3. Создайте модуль вывода.
4. Выполните задачу и осуществите проверку правильности передачи информации из модуля ввода в модуль вывода. Сохраните программу.
5. Проведите второй этап декомпозиции Вашей задачи. Уточните функциональную модель разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.
6. Проведите второй этап декомпозиции Вашей задачи. Уточните модель потока данных разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.

*Вариант 3*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех.

1. Скорректируйте программу, созданную на предыдущем лабораторном практикуме. Замените заглушки (операторы типа: Writeln('Блок ввода'); , Writeln('Блок запросов');) на обращение к модулям ввода и вывода.
2. Создайте модуль ввода.
3. Создайте модуль вывода.
4. Выполните задачу и осуществите проверку правильности передачи информации из модуля ввода в модуль вывода. Сохраните программу.
5. Проведите второй этап декомпозиции Вашей задачи. Уточните функциональную модель разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.
6. Проведите второй этап декомпозиции Вашей задачи. Уточните модель потока данных разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.

*Вариант 4*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой-участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде-участнице, а также упорядоченный список в соответствии с набранным количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5.

1. Скорректируйте программу, созданную на предыдущем лабораторном практикуме. Замените заглушки (операторы типа: Writeln('Блок ввода'); , Writeln('Блок запросов');) на обращение к модулям ввода и вывода.
2. Создайте модуль ввода.
3. Создайте модуль вывода.
4. Выполните задачу и осуществите проверку правильности передачи информации из модуля ввода в модуль вывода. Сохраните программу.
5. Проведите второй этап декомпозиции Вашей задачи. Уточните функциональную модель разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.
6. Проведите второй этап декомпозиции Вашей задачи. Уточните модель потока данных разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.

*Вариант 5*

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков.

1. Скорректируйте программу, созданную на предыдущем лабораторном практикуме. Замените заглушки (операторы типа: Writeln('Блок ввода'); , Writeln('Блок запросов');) на обращение к модулям ввода и вывода.
2. Создайте модуль ввода.
3. Создайте модуль вывода.
4. Выполните задачу и осуществите проверку правильности передачи информации из модуля ввода в модуль вывода. Сохраните программу.
5. Проведите второй этап декомпозиции Вашей задачи. Уточните функциональную модель разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.
6. Проведите второй этап декомпозиции Вашей задачи. Уточните модель потока данных разрабатываемого комплекса. Создайте схему с помощью средств Open Office.org Write.

## Лабораторный практикум № 9 Структурный подход к разработке программного обеспечения. Тестирование и отладка программного обеспечения

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 4.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить методы тестирования и отладки на практическом примере.

***Вводная часть***

*Методы тестирования и отладки*

*Тестирование* – процесс выполнения программ с целью обнаружения факта наличия ошибок.

*Отладка* – процесс локализации и устранения ошибок.

Процессы тестирования и отладки схематически могут быть представлены следующим образом (рисунок 23).

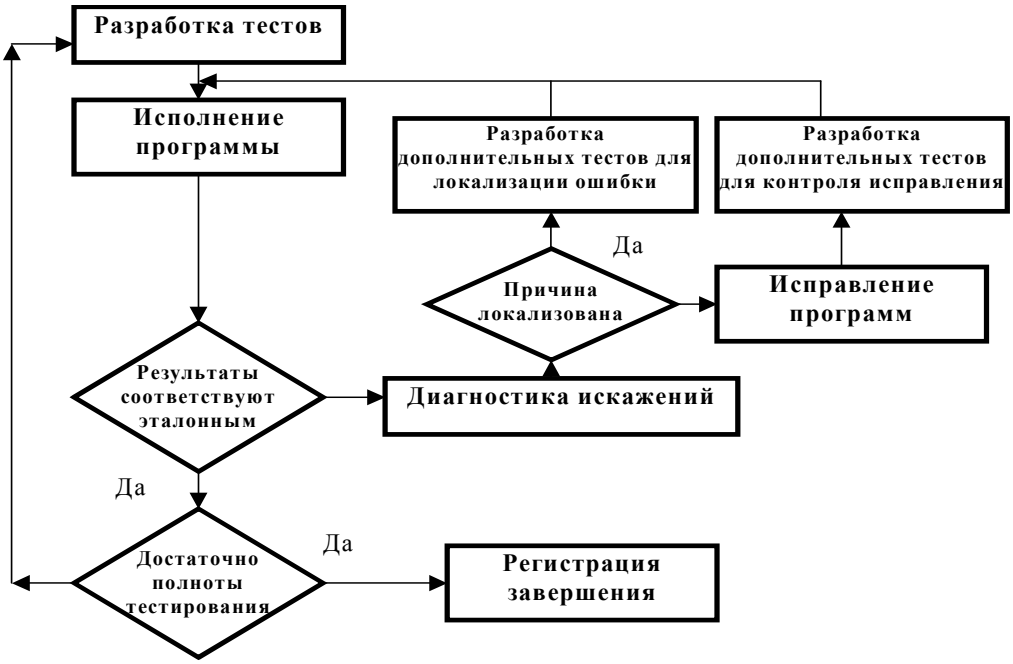



Рисунок 23. Процессы тестирования и отладки

Тестирование начинается с разработки множества тестов и их исполнения на основе одной из выбранных методик. Подготовка дополнительных тестов потребуется при недостаточной полноте тестирования, невозможности локализовать проблему с помощью имеющихся тестов и необходимости выполнить контроль сделанного исправления.

Существуют две основные стратегии тестирования.

Тестирование программы как *черного ящика,* при котором программа рассматривается как объект, внутренняя структура которого неизвестна.

Тестирование программы как *прозрачного (белого) ящика* подразумевает знание исходного кода программы и полный доступ к нему.

При тестировании по типу «черного ящика» тесты демонстрируют:

- как выполняются функции программы;
- как принимаются исходные данные;
- как вырабатываются результаты;
- как сохраняется целостность внешней информации.

При тестировании «черного ящика» рассматриваются системные характеристики программ, игнорируется их внутренняя логическая структура.

Для тестирования программ методом «черного ящика» готовятся определенные группы тестов.

- Для тестирования классов эквивалентностей. Классы эквивалентности позволяют вместо большого количества тестов использовать лишь их небольшое подмножество. Каждый тест представляет набор тестов, на которых программа ведет себя одинаково. Существует два типа классов эквивалентностей:

– класс корректных тестовых случаев, отражающих типичную «нормальную» ситуацию;

– класс тестов, содержащих ненормальную ситуацию, т.е. описывающих ситуацию, которой быть не должно.

- Для тестирования граничных значений.

- Для анализа причинно–следственных связей.
- Для тестирования тех утверждений, которые приводятся в документации.

При тестировании по типу «белого ящика» исследуются внутренние элементы и связи между ними. Объектом тестирования является не внешнее, а внутреннее поведение программы. Проверяется корректность построения всех элементов программы и правильность их взаимодействия друг с другом. Тестирование по принципу «белого ящика» характеризуется степенью, в какой тесты выполняют или покрывают логику (исходный текст) программы.

Наиболее важный принцип, относящийся к тестированию программ, состоит в том, чтобы думать об этой стадии еще на этапе написания программы. Следует постоянно задаваться вопросом: как будет тестироваться данный сегмент? Если ответ на вопрос о способе тестирования программы неясен, она должна быть либо переписана заново, либо разбита на модули.

Для составления тестов используются следующие источники:

- справочники;
- вычисления вручную;
- использование результатов, полученных при помощи другой программы.

Поскольку в процессе разработки приходится тестировать еще не завершенную программу, все подходы делятся на две группы.

*Тестирование сверху вниз***.** Применяется, если программа разрабатывается сверху вниз. В данном случае используются «заглушки» – фрагменты кода, имитирующие еще не написанные части программы.

*Тестирование снизу вверх***.** При этом, как правило, дополнительно должна быть создана программа – «драйвер», организующая взаимодействие уже написанных модулей.

Если процесс тестирования показал, что программа работает неправильно, то начинается процесс отладки. В процессе отладки локализуется ошибка.

Ошибки, возникающие в процессе проектирования и программирования приложений, можно подразделить на:

- синтаксические;
- времени выполнения;
- логические.

Редко удается ввести все операторы без единой ошибки. Даже самые опытные программисты допускают помарки при вводе операторов. Синтаксические ошибки, их также называют ошибками времени компиляции (compile-time error), наиболее легко устранимы. *Синтаксические ошибки* обнаруживаются компилятором автоматически. Сообщения о найденных ошибках отображаются в нижней части редактора. Сообщение описывает ошибку достаточно подробно, чтобы можно было понять ее причину.

При возникновении *ошибки времени выполнения* (run-time error) в программе среда разработки прерывает работу программы и на экране появляется диалоговое окно с сообщением о типе ошибки. Определить причину возникновения исключительной ситуации не сложно, но, исправив ошибку, работу программы приходится начинать с начала. Например, отсутствие функций приведения типов, неверное задание имен файлов.

*Логические ошибки* – это ошибки программиста при разработке алгоритма. Найти их сложнее, чем ошибки, возникающие во время выполнения программы.

Отладка программы – процесс творческий и плохо формализуемый. Тем не менее основной идее отладке можно придать вид следующего алгоритма.

- Следует начать с изучения уже доступных исходных и результирующих данных.
- Сформулировать некоторую гипотезу, которая объясняет получение таких результирующих данных.
- Подготовить новые исходные данные и провести эксперимент, который позволит доказать или опровергнуть гипотезу.

Для отладки программ в инструментальные среды программирования встраиваются специальные отладчики.

*Средства отладки PascalABC*

Отладка – это локализация ошибки и ее исправление. Для этого используются точки прерывания, пошаговое выполнение программы и просмотр ряда переменных на различных шагах выполнения программы.

В интегрированной среде PascalABC имеются следующие средства отладки:

- выполнение программы по операторам (без трассировки процедур и функций). Для выполнения одного шага (одной строки) программы следует нажать клавишу F8 или кнопку  ;
- выполнение программы по строкам с заходом в процедуры и функции (клавиша *F7* или кнопка  );
- выполнение программы до строки, на которой находится курсор. Для выполнения программы до данной строки следует установить на нее курсор и нажать клавишу F4 или кнопку  ;
- прекратить выполнение программы. Прервать программу, находящуюся в режиме пошагового выполнения, можно с помощью комбинации клавиш Ctrl–F2 или кнопки  ;
- окно отладки, в котором просматриваются значения переменных. Вызывается окно отладки пиктограммой  .

Эти возможности сосредоточены в пункте меню «Программа»: «Выполнить», «Завершить», «Шаг с входом в подпрограмму», «Шаг без входа в подпрограмму», «Выполнить до курсора», «Добавить выражение».

Если программа находится в режиме пошагового выполнения, то ее можно выполнить до конца, нажав F9.

Окно отладки позволяет просматривать во время пошагового исполнения программы значения переменных. По умолчанию оно располагается в правом верхнем углу окна редактора. Для добавления переменной или выражения в окно отладки следует нажать комбинацию клавиш Ctrl–F5 или кнопку  .

Можно также перетащить из редактора в окно отладки выделенное выражение или, при активном окне отладки, нажать клавишу *Ins*. Допускаются выражения, содержащие функции, в том числе и определенные в программе (такие функции не должны иметь побочного эффекта, т.е. не должны менять значения глобальных переменных).

При отладке обычно определяют точки останова. Выполнение программы приостанавливается в этих точках. В этот момент можно просмотреть значения интересуемых переменных, вычислить нужные выражения.

Опция «*Выполнить до курсора*» начинает или продолжает режим отладки исполняемой программы под управлением встроенного отладчика. Вначале осуществляются все действия по компиляции и компоновке программы, затем программа начинает работать обычным образом и останавливается перед выполнением первого оператора из строки, на которую указывает курсор. В этот момент экран возвращается в режим воспроизведения окна редактора, а строка с курсором выделяется цветным прямоугольником. В этом режиме доступны все средства встроенного отладчика. Можно перевести курсор к новой строке и вновь выбрать опцию «*Выполнить до курсора*»**.**

Значения переменных или выражений, которые необходимы при отладке, можно просмотреть в окне отладки.

***Практическая часть***

*Поиск ошибок в заданном примере с помощью средств отладки*

1. Ознакомьтесь с текстом программы:

```
Program Test;
Type Sball = record                  // Компонента файла
        Nom :Byte;                   // Номер группы
        SFiz, SInf, SMat : Real;     // Три оценки
        End;
Var d : Sball;
   f : file of Sball;
   s : array [1..10] of real;                  // Массив средних значений
   k : array [1..10] of Byte;        // Массив номеров групп
```

```
   i, j, q, c : byte;
   max : Real;
Begin
assign(f,'prob');
Reset (f);
For i := 1 To 10 Do                    // Записей в файле всего 10
  Begin
    Read(f,d);                         // Считываем компоненту файла
    S[i] :=  (d.SFiz + d.SInf + d.SMat)/3;   // Расчет среднего
    k[i] := d.nom;
    Writeln(k[i],'  ',s[i]);          // Вывод номера группы и средней оценки
  End;
   Close(f);
  For i := 1 To 9 do                   // Сортировка методом перестановок
   Begin
   Max := S[i]; q := i
{***} For j := i To 10 Do               // Ищем максимальное значение в массиве
      If s[j] > Max Then Begin
                  Max := s[j];
                  q := j;
                  End;
{**} s[q] := s[i] ; s[i] := Max;        // Переставляем текущий элемент с максимальным
   k[q] := k[i] ; k[i] := k[q];         // Переставляем номера групп
   end;
   For i := 1 To 10 Do Write (k[i],'  ');// Вывод номеров групп после упорядочения
   Writeln;
End.
```

Данная программа выполняет следующие действия:

- считывает из файла с именем «Proba» данные – номер группы и средние значения оценок по трем предметам;
- сортирует массив оценок в порядке убывания;
- выводит номера групп в соответствии с убыванием их средних значений.

Для сортировки массива используется метод перестановок. Номера групп в порядке убывания средних оценок формируется в массиве К.

Программа содержит три ошибки: синтаксическую, логическую и ошибку времени исполнения.

2. Запустите программу на выполнение.

Исправьте синтаксическую ошибку (в одном из операторов не хватает точки с запятой).

3. Запустите программу на выполнение.

Исправьте ошибку времени исполнения (имя файла записано неверно).

4. Запустите программу на выполнение.

Результат расчета не соответствует условию задачи. Следовательно, в программе имеется логическая ошибка.

5. Воспользуемся методами отладки приложения.

5.1. Установите курсор в положение оператора, отмеченного звездочками. Это оператор внутреннего цикла, в котором ищется максимум. В соответствии с алгоритмом метода перестановок максимальное будет найдено 9 раз.

5.2. Выведите на экран окно наблюдений. В окно наблюдений поместим две переменные Max и q (контекстное меню + добавить).

5.3. Запустите программу на выполнение до курсора (F4). В положении курсора программа остановится.

5.4. Далее выполните программу по шагам (F7). В окне наблюдений следите за значениями среднего балла и номера группы. При выходе из цикла средний балл принимает значение: 4.48, 4.27, …(см. промежуточный вывод в программе). Мы убедились, что максимальное значение среднего балла подсчитывается верно. Следовательно, ошибка может быть в операторах, расположенных ниже по программе.

5.5. Удалите переменные в окне наблюдений (контекстное меню + удалить).

5.6. Добавьте новые переменные S[i] и i.

5.7. Установите курсор в положение строки с двумя звездочками.

5.8. Выполните программу (F4, далее F7). Массив S[i] формируется правильно.

5.9. Следующим шагом необходимо проверить, как формируется массив k[i]. Задайте курсором точку останова в строке ниже, а в окно наблюдений добавьте переменные K[i] и i.

5.10. Массив К формируется неверно. Исправьте ошибку.

***Самостоятельная работа***

*Вариант 1*

*Текст программы Raspisanie.pas:*

```
Program Raspisanie;
Uses crt, vvod, Zapros;
Var    Fname: TStr;       {Имя файла}
      Reg : Byte;
Begin
  Repeat   {Цикл для организации обращения к блокам системы}
   Writeln('Выберите режим работы');
   Writeln(1,' ':10,'Ввод');
   Writeln(2,' ':10,'Запрос');
   Writeln(3,' ':10,'Конец работы');
   Writeln(' ':10,'Режим?');Readln(Reg);
   Case Reg Of
       1: Begin Vvodi(Fname); Delay(2000);End;
       2: Begin Zaprosi(Fname); Delay(2000);End;
       3: Begin Writeln('Конец работы'); Delay(2000);End;
   End;
 Until Reg = 3;
End.
```

*Модуль Vvod.pas:*

```
Unit Vvod;
Uses Crt;
{---------------------------------Vvod-------------------------------------}
{-------------------------Ввод исходных данных-------------------------}
Const
   Days: array[1..6] of String[2]=('Пн','Вт','Ср','Чт','Пт','Сб');
          {Массив дней недели}
Type TStr =String[20];
TPredm = array[1..20] of TStr;  {Тип данных для названий предметов}
TPrep =  array[1..20] of TStr;  {Тип данных для фамилий преподавателей}
TRasp =array [1..4] of TStr;    {Тип данных для файла расписания}
Var    Prep : TPrep;           {Массив фамилий преподавателей}
      Pred : TPredm;          {Массив предметов}
      R : TRasp;              {Компонента типизированного файла}
```

```
      f : file of TRasp;       {Файловая переменная}
      f1 : Text;               {Файловая переменная}
      Fname: TStr;       {Имя файла}
      Reg: Byte;
Procedure Vvodi (Var FName:TStr);
Var sss : TStr; i :Byte;
Begin
  Repeat
  Clrscr;
   Writeln('Выберите режим работы');
   Writeln(1,' ':10,'Файл предметов');
   Writeln(2,' ':10,'Файл преподавателей');
   Writeln(3,' ':10,'Файл расписания');
   Writeln(4,' ':10,'Конец работы');
   Writeln(' ':10,'Режим?');Readln(Reg);
   Case Reg Of
     1: Begin     Clrscr;
         Assign(f1, 'Predmet.dat');
         ReWrite(f1);
         For i:= 1 To 7 Do
            Begin
            Write(i,'--'); Readln(sss);
            Writeln(f1,sss);
            end;
        Close(f1);
       End;
     2: Begin      Clrscr;
        Assign(f1, 'Prepod.dat');
         ReWrite(f1);
         For i:= 1 To 7 Do
            Begin
            Write(i,'--'); Readln(sss);
            Writeln(f1,sss);
            end;
        Close(f1);
       End;
     3: Begin          Clrscr;
        Write ('Имя файла'); Readln(fName);
        Assign(f, FName);
         ReWrite(f);
         For i:= 1 To 14 Do
            Begin
            Write(i,'День---'); Readln(R[1]);
            Write(i,'Преподаватель---'); Readln(R[2]);
            Write(i,'Предмет---'); Readln(R[3]);
            Write(i,'Номер пары---'); Readln(R[4]);
            Write(f,R);
            end;
```

```
        Close(f);
       End;
    4: Begin Writeln('Конец работы'); Delay(2000);End;
  End;
    Clrscr;
 Until Reg = 4;
End;
End.
```

*Модуль Zapros.pas*

```
Unit Zapros;
Uses Crt;
{------------------------------Запрос 2-----------------------------------}
//  Запрос 2.
//Найти количество занятий, по заданному предмету в неделю.
//Исходной информацией для этого запроса является название предмета Z и файл,
//в котором хранится расписание. Если Z = R[3], то Kol = Kol +1,
//где R[3] - названия предметов, хранящихся в файле "расписание".
Type TStr =String[20];
TPredm = array[1..20] of TStr;  {Тип данных для названий предметов}
TPrep =  array[1..20] of TStr;  {Тип данных для фамилий преподавателей}
TRasp =array [1..4] of TStr;    {Тип данных для файла расписания}
Var    Prep : TPrep;           {Массив фамилий преподавателей}
      Pred : TPredm;           {Массив предметов}
      R : TRasp;
      b : Array [1..14,1..4] of TStr;  {Массив расписания}
      f : file of TRasp;       {Файловая переменная}
      f1 : Test;               {Файловая переменная}
      Fname: TStr;       {Имя файла}
Procedure Zaprosi(FName: String);
Var i, j, Kol : Byte; Z : TStr;
Begin
Clrscr;
//Считываем названия предметов
  Repeat
  Try
  Assign(f1, 'predmet');
    {Блок проверки наличия файла на диске}
  Reset(f1);
    Except
    If not(FileExists('predmet.dat')) then
      Writeln('Файла с именем "Predmet" не существует на диске!');
    End;
  Until FileExists('predmet.dat');
  For i := 1 To 7 Do Readln(f1, Pred[i]);
  For i := 1 To 7 Do Writeln(' ':10,Pred[i]);
  Kol := 7;
  Close(f1);
//Считываем расписание
```

```
 Repeat
   Write('Имя файла с результатами сессии?'); Readln(Fname);
  Try
   Assign(f, Fname);
   Reset(f);
   Except
  If not (FileExists(fname)) then
                Writeln('Файла с таким именем не существует на диске!');
   End;
 Until FileExists(Fname);
 For i := 1 To 14 Do
 Begin
  Read(f,R);
  For j:= 1 To 4 Do  b[i,j] := R[j]; //Переписали расписание в массив b
End;
 Close (f);
// Запрос 1  Количество занятий по данному предмету
 Write('Введите название предмета ');Readln (Z);
 For i := 1 To 14 Do
   For j :=1 To 4 Do
    If b[i,j] = Z Then Kol := Kol +1;
 Writeln ('По предмету ', Z, ' проводится ', Kol, ' занятий.');
End;
End.
```

Отладить программу Raspisanie.pas, состоящую из основной программы и двух модулей: Vvod.pas и Zapros.pas. Ошибки сосредоточены в модуле Zapros, поэтому выполнять необходимо только ту часть программы, которая относится к запросам. Модуль Zapros.pas решает часть задачи, поставленной в первом лабораторном практикуме.

«Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота)».

1. Ознакомьтесь с программой. Прочитайте содержание запроса в комментариях модуля Zapros.pas. Имя файла, в котором хранятся данные по расписанию – Rasp.dat. Названия предметов хранятся в файле Predmet.dat.
2. Найдите синтаксическую ошибку в программе.
3. Найдите ошибку времени исполнения.
4. Найдите логическую ошибку в программе, воспользовавшись средствами отладки PascalABC.

*Вариант 2*

*Текст программы Otdel.pas:*

```
Program Otdel;
Uses crt, Vvod, Zapros;
Begin
  Repeat   {Цикл для организации обращения к блокам системы}
   Writeln('Выберите режим работы');
   Writeln(1,' ':10,'Ввод');
   Writeln(2,' ':10,'Запрос');
   Writeln(3,' ':10,'Конец работы');
```

```
  Writeln(' ':10,'Режим?');Readln(Reg);
  Case Reg Of
      1: Begin Vvodi(Fname); Delay(2000);End;
      2: Begin Zaprosi(Fname); Delay(2000);End;
      3: Begin Writeln('Конец работы'); Delay(2000);End;
  End;
 Until Reg = 3;
End.
```

*Модуль Otdel.pas:*

```
Unit Vvod;
{---------------------------------Vvod-------------------------------------}
{--------------------------Ввод результатов сессии-------------------------}
Uses crt;
Type TStr =String[20];
    TFIO = record
          Fam, Name1, Name2 : TStr;
          End;
    TBDay = record
           dd : 1 .. 31;
           mm : 1 .. 12;
           gg : 1920 .. 2020;
           End;
     TAdres = Record
             Add1, Add2 : TStr;
             Dom, Kvar : Integer;
             End;
     TZap = Record
           Name : TFio;
           BDay : TBDay;
           Adr : TAdres;
           Rate : Real
           End;
Var    Pers : TZap;            {Компонента типизированного файла}
      f : file of TZap;       {Файловая переменная}
      Fname: String[20];      {Имя файла}
Procedure Vvodi (Var FName:TStr);
Var i : Byte;
Begin
  Writeln('Имя файла данных по сотрудникам?'); Readln(Fname);
       Clrscr;
       Assign(f, FName);
       ReWrite(f);
       For i:= 1 To 7 Do
         Begin
            Write(i,'Фамилия---'); Readln(Pers.Name.Fam);
            Write(i,'Имя---'); Readln(Pers.Name.Name1);
            Write(i,'Отчество---'); Readln(Pers.Name.Name2);
          Writeln('Адрес');
```

```
            Write(i,'Город---'); Readln(Pers.Adr.Add1);
            Write(i,'Улица---'); Readln(Pers.Adr.Add2);
            Write(i,'Дом---'); Readln(Pers.Adr.Dom);
            Write(i,'Квартира---'); Readln(Pers.Adr.Kvar);
          Writeln('День рождения');
            Write(i,'День---'); Readln(Pers.Bday.dd);
            Write(i,'Месяц---'); Readln(Pers.Bday.mm);
            Write(i,'Год---'); Readln(Pers.Bday.gg);
          Write(i,'Зарплата---'); Readln(Pers.rate);
            Write(f,Pers);
        end;
     Close(f);
 End;
End.
```

*Модуль Zapros.pas:*

```
Unit Zapros;
// Запрос 4
//Поиск минимального. Для всех записей, если Мин > "Зарплата", то Мин = "Зарплата".
//Если "Зарплата" = Мин, вывести "Фамилия".
Uses Crt;
{---------------------------Отладка запроса 4-----------------------------}
Type TStr =String[20];
    TFIO = record
          Fam, Name1, Name2 : TStr; // ФИо
          End;
    TBDay = record
           dd : 1 .. 31;           // День рождения
           mm : 1 .. 12;           // Месяц
           gg : 1920 .. 2020;      // Год
           End;
     TAdres = Record
            Add1, Add2 : TStr;     // Город, улица
            Dom, Kvar : Integer;   // Дом, квартира
            End;
     TZap = Record
           Name : TFio;       // Фамилия
           BDay : TBDay;      // День рождения
           Adr : TAdres;      // Адрес
           Rate : Real        // Заработная плата
           End;
Var    Pers : TZap;            {Компонента типизированного файла}
      f : file of TZap;        {Файловая переменная}
      Fname: String[20];       {Имя файла}
      Reg: Byte;               {Вспомогательная переменная для организации меню}
Procedure Zaprosi(FName: TStr);
Var i, k : byte; Min : Real;
   Fio : Array [1..6] of TStr;
   z : Array [1..7] of Real;
```

```
Begin
 Repeat
   Write('Имя файла данных?'); Readln(Fname);
   // Проверка наличия файла на диске
  Try
   Asign(f, 'Fname');
   Reset(f);
   Except
  If not (FileExists(fname)) then
                Writeln('Файла с таким именем не существует на диске!');
   End;
 Until FileExists(Fname);
Clrscr;
// Считываем данные из файла
      For i:= 1 To 7 Do
       Begin
         Read(f,Pers);
         Fio [1] := Pers.Name.Fam; // Формируем массив фамилий
         Z[i] :=  Pers.rate;       // Формируем массив заработанных плат
       end;
     Close(f);
     // Запрос 4. Список сотрудников, имеющих минимальную заработную плату
     Min := Z[1];
     For i := 2 To 7 Do
        If  Z[i] < Min Then  Min := Z[i];
     For i := 1 To 7 Do
        If Z [i] = Min Then Writeln(Fio[i]);
End;
End.
```

Отладить программу Otdel.pas, состоящую из основной программы и двух модулей: Vvod.pas и Zapros.pas. Модуль Zapros.pas решает часть задачи, поставленной в первом лабораторном практикуме:

«Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату».

1. Ознакомьтесь с программой. Прочитайте содержание запроса в комментариях модуля Zapros.pas. Имя файла, в котором хранятся данные по расписанию – Otdel.dat.
2. Найдите синтаксическую ошибку в программе.
3. Найдите ошибку времени исполнения.
4. Найдите логическую ошибку в программе, воспользовавшись средствами отладки PascalABC.

*Вариант 3*

*Текст программы Rielte.pas:*

```
Program Rielte;
Uses crt, Vvod, Zapros;
{----------------------------Риэлтовская фирма-------------------------------}
Var
     Fname: String[20];      {Имя файла}
     Reg: Byte;              {Вспомогательная переменная для организации меню}
```

```
Begin
  Repeat   {Цикл для организации обращения к блокам системы}
   Writeln('Выберите режим работы');
   Writeln(1,' ':10,'Ввод');
   Writeln(2,' ':10,'Запрос');
   Writeln(3,' ':10,'Конец работы');
   Writeln(' ':10,'Режим?');Readln(Reg);
   Case Reg Of
       1: Begin Vvodi(Fname); Delay(2000);End;
       2: Begin Zaprosi(Fname); Delay(2000);End;
       3: Begin Writeln('Конец работы'); Delay(2000);End;
   End;
 Until Reg = 3;
End.
```

*Модуль Vvod.pas:*

```
Unit Vvod;
Uses crt;
{---------------------------------Vvod-------------------------------------}
{-------------------------Ввод исходной информации-------------------------}
Type   TStr = String [20];
THome = record
            Sq_O, Sq_G : real;
            Kol, Pr_S, Pr_L, Pr_D : Integer;
            Stoim : Real;
           End;
Var    Pers : THome;            {Компонента типизированного файла}
      f : file of THome;       {Файловая переменная}
      Fname: TStr;      {Имя файла}
Procedure Vvodi (Var FName:TStr);
Var i : Byte;
Begin
   Writeln('Имя файла данных по жилым помещениям?'); Readln(Fname);
      Clrscr;
      Assign(f, FName);
      ReWrite(f);
      For i:= 1 To 7 Do
        Begin
           Write(i,'Общая площадь---'); Readln(Pers.Sq_o);
           Write(i,'Жилая площадь---'); Readln(Pers.Sq_G);
           Write(i,'Количество комнат---'); Readln(Pers.Kol);
           Write(i,'Сан. узел (совместный 0, раздельный 1)---'); Readln(Pers.Pr_S);
           Write(i,'Лоджия---'); Readln(Pers.Pr_L);
           Write(i,'Тип дома (панельный -0, кирпичный - 1)---'); Readln(Pers.Pr_D);
           Write(i,'Стоимость (млн)---'); Readln(Pers.Stoim);
           Write (f,Pers);
       end;
     Close(f);
End;
```

```
End.
```

*Модуль Zapros.pas:*

```
Unit Zapros;
Uses Crt;
{----------------------------Запрос 3---------------------------------}
//Запрос 3
//Сортировка по убыванию поля "Стоимость". Применить метод перестановок.
Type TStr = String [20];
THome = record
          Sq_O, Sq_G : real;          // Общая и жилая площадь
          //количество комнат - Kol
          // признак санузла -  Pr_S
          // признак лоджии  -  Pr_L
          // признак дома    -  Pr_D
          Kol, Pr_S, Pr_L, Pr_D : Integer;
          Stoim : Real;
         End;
Var    Pers : THome;           {Компонента типизированного файла}
      c : array [1..7] of Thome;
      f : file of THome;       {Файловая переменная}
      Fname: TStr;      {Имя файла}
Procedure Zaprosi(FName: TStr);
Var k : array [1..7] of Byte;
   i, j, q : Byte;
   z : array [1..7] of Real;
   Max : Real;
Begin
// Запрос 3. Сортировка по убыванию стоимости квартиры
Clrscr;
// Проверка существования файла на диске
Repeat
    Write('Имя файла данных?'); Readln(Fname);
  Try
   Assign(f, Fname); //Reset (f);
  Except
  If not (FileExists(fname)) then
                Writeln('Файла с таким именем не существует на диске!');
   End;
 Until FileExists(Fname);
 For i:= 1 To 7 Do k[i] := i;
For i:= 1 To 7 Do
        Begin
          Read(f,c[i]);
          With c[i] do
           Z[i] :=  c[i].stoim; // Формирование массива стоимостей
        end;
Close(f);
 // Сортировка методом перестановок
```

```
For i:= 1 To 7 Do
Begin
 Max := z[i]; q := i;
 For i:= 1 To 7 Do
  If  z[j] > max Then
              Begin
               Max := Z[j]; q := j;
              End;
 z[q] := z[i]; z[i] := Max;
 x := k[i]; k[i] := k[q]; k[q] := x;
End;
// Вывод результата
For i := 1 To 7 Do
 Begin
   j := k[i];
   With c[j] do
   Writeln (Sq_O,'  ', Sq_G,'  ', Kol, '  ',Pr_S,'  ', Pr_L,'  ', Pr_D, '  ',Stoim);//
 End;
End;
End.
```

Отладить программу Rielte.pas, состоящую из основной программы и двух модулей: Vvod.pas и Zapros.pas. Ошибки сосредоточены в модуле Zapros, поэтому выполнять необходимо только ту часть программы, которая относится к запросам. Модуль Zapros.pas решает часть задачи, поставленной в первом лабораторном практикуме:

«Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех».

1. Ознакомьтесь с программой. Прочитайте содержание запроса в комментариях модуля Zapros.pas. Имя файла, в котором хранятся данные по расписанию – Rielter.dat.
2. Найдите синтаксическую ошибку в программе.
3. Найдите ошибку времени исполнения.
4. Найдите логическую ошибку в программе, воспользовавшись средствами отладки PascalABC.

*Вариант 4*

*Текст программы Olimpic.pas:*

```
Program Olimpic;
Uses crt, Vvod, Zapros;
{--------------------------------Олимпиада-------------------------------------}
Type
Var
    Fname: String[20];      {Имя файла}
    Reg: Byte;              {Вспомогательная переменная для организации меню}
Begin
 Repeat   {Цикл для организации обращения к блокам системы}
  Writeln('Выберите режим работы');
  Writeln(1,' ':10,'Ввод');
  Writeln(2,' ':10,'Запрос');
```

```
  Writeln(3,' ':10,'Конец работы');
  Writeln(' ':10,'Режим?');Readln(Reg);
  Case Reg Of
      1: Begin Vvodi(Fname); Delay(2000);End;
      2: Begin Zaprosi(Fname); Delay(2000);End;
      3: Begin Writeln('Конец работы'); Delay(2000);End;
  End;
 Until Reg = 3;
End.
```

*Модуль Vvod.pas:*

```
Unit Vvod;
Uses Crt;
{---------------------------------Vvod-------------------------------------}
{-------------------------Ввод исходной информации-------------------------}
Type TStr = String [20];
    TMedali = record
            Strana : TStr;
            Kol_Z, Kol_S, Kol_B : Integer;
            End;
Var    Pers : TMedali;            {Компонента типизированного файла}
      f : file of TMedali;       {Файловая переменная}
      Fname: TStr;       {Имя файла}
Procedure Vvodi (Var FName:TStr);
Var i : Byte;
Begin
   Writeln('Имя файла данных по командам?'); Readln(Fname);
      Clrscr;
      Assign(f, FName);
      ReWrite(f);
      For i:= 1 To 7 Do
       Begin
          Write(i,'Команда---'); Readln(Pers.Strana);
          Write(i,'Золото---'); Readln(Pers.Kol_Z);
          Write(i,'Серебро---'); Readln(Pers.Kol_S);
          Write(i,'Бронза---'); Readln(Pers.Kol_B);
          Write(f,Pers);
       end;
     Close(f);
End;
End.
```

*Модуль Zapros.pas:*

```
Unit Zapros;
Uses Crt;
{-------Запрос 3. Сортировка в порядке убывания количества очков----------}
//Запрос 3. Вывести упорядоченный список команд в
//соответствии с набранным количеством очков.
// Определяем количество очков в каждой команде:
// Zi = Kol_zi*7 + Kol_si *6 + Kol_bi*5.
```

```
// Осуществляем сортировку массива Z методом перестановок.
Type TStr = String [20];
    TMedali = record
            Strana : TStr; //  Название команды
            // Количество золотых медалей     -  Kol_Z
            // Количество серебрянных медалей -  Kol_S
            // Количество бронзовых медалей   -  Kol_B
            Kol_Z, Kol_S, Kol_B : Integer;
            End;
Var    Pers : TMedali;            {Компонента типизированного файла}
      f : file of TMedali;       {Файловая переменная}
      Fname: String[20];       {Имя файла}
Procedure Zaprosi(FName: TStr);
Var i, j, q, x : Byte; Max : Integer;
   Naz : array [1...7] of TStr;
   Z : array [1..7] of Integer;
   k : array [1..7] of Byte;
Begin
// Запрос 3. Сортировка в порядке убывания количества очков
Clrscr;
// Проверка наличия файла на диске
Repeat
   Write('Имя файла данных?'); Readln(Fname);
  Try
   Assign(f, 'Fname');
   Reset(f);
   Except
  If not (FileExists(fname)) then
                Writeln('Файла с таким именем не существует на диске!');
   End;
 Until FileExists(Fname);
 For i:= 1 To 7 Do k[i] := i; // Подготавливаем массив для индексов
For i:= 1 To 7 Do
        Begin
          Read(f,Pers);
          With Pers do
          Begin
           Naz[i] := Strana;
            //Количество набранных очков
           Z[i] :=  Kol_Z *7 + Kol_S * 6 + Kol_B * 5;
          End;
        end;
Close(f);
// Сортировка методом перестановок
For i := 1 To 7 Do k[i] := i;
For i := 1 To 7 Do
 Begin
  Max := z[i]; q := i;
```

```
 For j := i To 7 Do
  If z[i] > Max Then Begin
                Max := Z[i]; q := i;
                End;
 Z[q] := Z[i]; Z[i] := max;
 x := k[i]; k[i] := k[q]; k[q] := x;
End;
// Вывод результатов
 For i := 1 To 7 Do
 Begin
  j := k[i];
  Writeln(Naz[j]);
 End;
End;
End.
```

Отладить программу Olimpic.pas, состоящую из основной программы и двух модулей: Vvod.pas и Zapros.pas. Ошибки сосредоточены в модуле Zapros, поэтому выполнять необходимо только ту часть программы, которая относится к запросам. Модуль Zapros.pas решает часть задачи, поставленной в первом лабораторном практикуме:

«Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой-участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде-участнице, а также упорядоченный список в соответствии с набранным количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5».

1. Ознакомьтесь с программой. Прочитайте содержание запроса в комментариях модуля Zapros.pas. Имя файла, в котором хранятся данные по расписанию – Olimpic.dat.
2. Найдите синтаксическую ошибку в программе.
3. Найдите ошибку времени исполнения.
4. Найдите логическую ошибку в программе, воспользовавшись средствами отладки PascalABC.

*Вариант 5*

*Текст программы Post.pas:*

```
Program Post;
Uses crt, Vvod, Zapros;
Var
    Fname: String[20];      {Имя файла}
    Reg: Byte;              {Вспомогательная переменная для организации меню}
Begin
 Repeat   {Цикл для организации обращения к блокам системы}
  Writeln('Выберите режим работы');
  Writeln(1,' ':10,'Ввод');
  Writeln(2,' ':10,'Запрос');
  Writeln(3,' ':10,'Конец работы');
  Writeln(' ':10,'Режим?');Readln(Reg);
  Case Reg Of
     1: Begin Vvodi(Fname); Delay(2000);End;
     2: Begin Zaprosi(Fname); Delay(2000);End;
     3: Begin Writeln('Конец работы'); Delay(2000);End;
```

```
  End;
 Until Reg = 3;
End.
```

*Модуль Vvod.pas:*

```
Unit Vvod;
Uses Crt;
{---------------------------------Vvod-------------------------------------}
{-------------------------Ввод исходной информации-------------------------}
Type TStr = String [20];
    TIzdanie = record
            Izdan : TStr;
            Pr_V, Nom : Integer;
            Name : TStr;
            End;
Var    Pers : TIzdanie;            {Компонента типизированного файла}
      f : file of TIzdanie;       {Файловая переменная}
      Fname: TStr;       {Имя файла}
Procedure Vvodi (Var FName:TStr);
Var i : Byte;
Begin
   Writeln('Имя файла данных по изданиям?'); Readln(Fname);
      Clrscr;
      Assign(f, FName);
      ReWrite(f);
      For i:= 1 To 13 Do
        Begin
           Write(i,'Издание---'); Readln(Pers.Izdan);
           Write(i,'Выпуск---'); Readln(Pers.Pr_V);
           Write(i,'Участок---'); Readln(Pers.Nom);
           Write(i,'Фамилия---'); Readln(Pers.Name);
           Write(f,Pers);
       end;
     Close(f);
End;
End.
```

*Модуль Zapros.pas:*

```
Unit Zapros;
Uses Crt;
{---------------------------------Запрос 1-----------------------------------}
// Вводятся: месяц (M) и название (name).
// Осуществляем перебор записей и проверку по двум признакам
// "Месяц" = Mes и "Название" = Nazv.
// Одновременно проверяем по "признаку" существует ли данное издание в этом месяце:
// Если "признак" = 2, то проверка "Месяц" - нечетный.
// Если "признак" = 3, то проверка "Месяц" = 1 или "Месяц" = 7.
// Если все условия выполняются, то Kol = Kol +1.
Const mm : array [1..12] Of String [10] = ('Январь', 'Февраль',
                          'Март','Апрель','Май','Июнь','Июль','Август',
```

```
                        'Сентябрь','Октябрь','Ноябрь','Декабрь');
Type TStr = String [20];
    TIzdanie = record
            Izdan : TStr;       // Название издания
            Pr_V, Nom : Integer; // Признак выпуска, номер участка
            Name : TStr;        // Имя подписчика
            End;
Var    Pers : TIzdanie;            {Компонента типизированного файла}
      f : file of TIzdanie;       {Файловая переменная}
      Fname: TStr;      {Имя файла}
Procedure Zaprosi(FName: TStr);
Var i,k, Mes : Byte; Nazv : TStr;
Begin
Clrscr;
// Запрос 1. По месяцу и названию определить количество экземпляров,
// подлежащих доставке
Clrscr;
// Проверка наличия файла на диске
Repeat
   Write('Имя файла данных?'); Readln('Fname');
  Try
   Assign(f, 'Fname');
   Reset(f);
   Except
  If not (FileExists(fname)) then
                Writeln('Файла с таким именем не существует на диске!');
   End;
 Until FileExists(Fname);
 Write ('Месяц? (введите номер месяца)'); Readln(Mes); // Номер месяца
 Write ('Название? '); Readln (Nazv);                // Название издания
  // Считываем из файла
  k := 0;
For i:= 1 To 7 Do
        Begin
          Read(f,Pers);
          With Pers do
          Begin
//          Writeln(Izdan,'  ', Pr_v,'   ', Nom,'   ', Name);
           If Izdan = Nazv Then                    //Проверка условия
                Begin
                  If (Pr_V = 1) Then k := k + 1;
                  If (Pr_V = 2)and ((mes mod 2)<> 0 ) Then
                                        k := k + 1;
                  If (Pr_V = 3)and ((mes = 1) or (mes = 7) ) Then
                                        k := k + 1;
                End;
          End;
        end;
```

```
Close(f);
// Вывод результата
Writeln ('В ', mm[mes],' необходимо доставить ', k, ' экземпл. ', Nazv);
End;
End.
```

Отладить программу Post.pas, состоящую из основной программы и двух модулей: Vvod.pas и Zapros.pas. Ошибки сосредоточены в модуле Zapros, поэтому выполнять необходимо только ту часть программы, которая относится к запросам. Модуль Zapros.pas решает часть задачи, поставленной в первом лабораторном практикуме:

«Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков».

1. Ознакомьтесь с программой. Прочитайте содержание запроса в комментариях модуля Zapros.pas. Имя файла, в котором хранятся данные по расписанию – Post.dat.
2. Найдите синтаксическую ошибку в программе.
3. Найдите ошибку времени исполнения.
4. Найдите логическую ошибку в программе, воспользовавшись средствами отладки PascalABC.

## Лабораторный практикум № 10 Оценка качества программного продукта. Использование библиотеки поддержки разработки

***Продолжительность: 90 минут.***

***Дисциплина «Технология программирования». Юнита 4.***

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

***Цель занятия:*** изучить основные характеристики качества программных продуктов; изучить на практическом примере процесс подключения стандартных программ из библиотеки поддержки разработки.

***Вводная часть***

*Основные понятия и характеристики качества программных продуктов*

Принципиальной особенностью программного средства (ПС) является невозможность выделения единственного критерия качества, полностью характеризующего данное ПС, его функциональные и конструктивные особенности. Однако в зависимости от стадии в жизненном цикле ПС и целей исследования качества программного средства на роль доминирующего или обобщающего критерия может быть выбран один или несколько критериев качества программного обеспечения.

Сущность измерения показателей качества ПС в общем виде сводится к получению информации о фактическом состоянии ПС и сопоставлении этой информации с заранее установленными требованиями, нормами и критериями. При этом устанавливается соответствие или несоответствие фактического состояния качества ПС требуемому (ожидаемому). Далее принимаются конкретные решения по обеспечению качества ПС.

Можно условно разделить все критерии качества на функциональные и конструктивные. Функциональные критерии отражают основную специфику применения и степень соответствия программ их целевому назначению. Конструктивные критерии качества программ более или менее инвариантны к их целевому назначению и основным функциям.

Функциональные критерии весьма различны и соответствуют разнообразию целевого назначения функций и области применения ПС. Для каждого ПС или группы ПС могут быть свои уникальные функциональные критерии качества. Например, для программ обучения выделяются показатели наглядности, вариантности и т.п.

Выбор характеристик и оценка качества программных средств – лишь одна из задач в области обеспечения качества продукции. Комплексное решение задач обеспечения качества программных средств предполагает разработку и внедрение той или иной системы управления качеством. В мировой практике наибольшее распространение получила система, основанная на международных стандартах серии ISO 9000, включающей десяток с лишним документов, в том числе стандарт, регламентирующий обеспечение качества ПО (ISO 9000/3).

*Характеристики качества (ISO 9126-1)*

*Функциональные возможности* – способность программного средства обеспечивать решение задач, удовлетворяющих сформулированные потребности заказчиков и пользователей при применении комплекса программ в заданных условиях.

*Функциональная пригодность* – набор и описания субхарактеристики и ее атрибутов, определяющие назначение, номенклатуру, основные, необходимые и достаточные функции программного средства, соответствующие техническому заданию и спецификациям требований заказчика или потенциального пользователя.

*Правильность (корректность)* – способность программного средства обеспечивать правильные или приемлемые для пользователя результаты и внешние эффекты.

*Способность к взаимодействию* – свойство программных средств и их компонентов взаимодействовать с одной или большим числом компонентов внутренней и внешней среды.

*Защищенность* – способность компонентов программного средства защищать программы и информацию от любых негативных воздействий.

*Надежность* – обеспечение комплексом программ достаточно низкой вероятности отказа в процессе функционирования программного средства в реальном времени.

*Эффективность* – свойства программного средства, обеспечивающие требуемую производительность решения функциональных задач, с учетом количества используемых вычислительных ресурсов в установленных условиях.

*Практичность (применимость)* – свойства программного средства, обусловливающие сложность его понимания, изучения и использования, а также привлекательность для квалифицированных пользователей при применении в указанных условиях.

*Сопровождаемость* – приспособленность программного средства к модификации и изменению конфигурации и функций.

*Мобильность* – подготовленность программного средства к переносу из одной аппаратно-операционной среды в другую.

*Изучение возможностей библиотеки стандартных процедур*

Использование защищенного ввода и наглядного вывода значительно улучшают качество программного средства.

При использовании технологии нисходящего структурного программирования имеется возможность включать в разрабатываемые программные комплексы модули, созданные ранее. Использование готовых программных модулей значительно повышает надежность программного комплекса, упрощает процесс его разработки, сокращает время разработки и снижает стоимость самой разработки.

Оттестированные программные модули, которые могут использоваться в других программах без каких–либо изменений или с небольшими изменениями, называются стандартными программными модулями или *стандартными процедурами*, а алгоритмы, которые они реализуют, – *стандартными алгоритмами*. Каждый программист, как правило, со временем сам создает значительное количество процедур, с помощью которых решаются общие для многих программ задачи (элементы оформления программ, организация интерфейса пользователь – компьютер, средства контроля и защиты от ошибок пользователя и др.). Эти процедуры, являясь стандартными процедурами, могут составить постоянно пополняющуюся личную библиотеку программиста.

Библиотека поддержки разработки (БПР) служит для хранения и использования всех стандартных программ. БПР также предназначена для организации хранения текущего состояния проекта и архивов.

Обучающимся предлагается библиотека стандартных процедур, которые предназначены для разработки дружественных к пользователю и защищенных программных комплексов.

В данном лабораторном практикуме предоставляется возможность использования стандартных процедур для защиты ввода и оформления вывода (таблица 1).

Таблица 1. Библиотека стандартных процедур

| Название п/п | Назначение | Заголовок | Параметр |
|---|---|---|---|
| Menu | Программа вывода горизонтального меню | Procedure menu_1 (Ypos: Byte); | Ypos – позиция вывода меню по вертикали |
| ReadReal | Процедура ввода вещественного числа с контролем ввода | Procedure ReadReal (Var RealVar : Real; Var CodIOR: Integer); | RealVar – вводимое число, CodIOR – код ошибки |
| ReadInt | Процедура ввода целого числа с контролем ввода | Procedure ReadInt (Var IVar : Integer; Var CodIOR: Integer); | IVar – вводимое число, CodIOR – код ошибки |
| ReadRuss | Процедура ввода символов кириллицы | Procedure ReadRuss( Var Stroka: String; Var CodIor: Integer); | Stroka – вводимая строка; CodIor – код ошибки |
| goingout | Процедура приостанавливает работу программы и позволяет выводить требуемое сообщение желаемого цвета и располагать его в нужном месте экрана | Procedure Goingout (Text : String; X,Y : Integer); | Text – текст сообщения. ColorText – цвет текста. X, Y – координаты вывода сообщения |
| goingouc | Процедура задержки экрана. Выводится подсказка о дальнейших действиях пользователя. Устанавливаются цвета фона и символов | Procedure GoingOutColor ( Text : String; ColorText, ColorBackGround: Byte; x, y: integer); | Text – текст сообщения. ColorText – цвет символов. ColorBackGround – цвет фона. X, Y – координаты вывода сообщения |
| frame | Процедура построения рамки для выделения элементов, выводимых на экран | Procedure Frame (x1, y1, x2, y2, FrameColor, TypeLine : Integer); | x1, y1, x2, y2 – координаты левого верхнего и правого нижнего углов. FrameColor – цвет рамки. TypeLine – тип рамки: 1 – одинарная линия, 2 – двойная линия |
| Middle | Процедура выводит на экран строку текста, центрируя ее. Длина выводимой строки не должна превышать 80 символов | Procedure Middle (Stroka : String; Y: integer); | Stroka – выводимая строка. Y – координа вывода по вертикали |
| ColrStr | Вывод строки с центрированием и изменением яркости | Procedure ColorStr (Stroka : String; Y: integer); | Stroka – выводимая строка. Y – координа вывода по вертикали |
| Zast_cs | Процедура вывода заставки, в которой выводимые сообщения изменяют цвет | Procedure Zast_cs; | Без параметров |

***Практическая часть***

*Пример использования стандартных подпрограмм*

Подключение стандартных подпрограмм рассмотрим на примере. Напомним условие задачи, рассмотренной на первых лабораторных практикумах:

Разработать программный комплекс по обработке результатов сессии на курсе. Программный комплекс позволяет получать следующие сведения:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;

- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

Входными данными в этой задаче являются:

1. Массив названий предметов, по которым тестировались обучающиеся курса.
2. Информация по каждому обучающемуся курса:

a) номер группы;
b) фамилия;
c) имя;
d) отчество;
e) оценки по предметам сессии (массив).

Для защиты ввода можно использовать стандартные процедуры ReadInt и ReadRuss.

ReadRuss – процедура ввода символов кириллицы. Эта процедура не позволяет использовать никакие другие символы, кроме кириллицы. Переменные строкового типа необходимо вводить, используя эту процедуру, а именно: названий предметов, фамилия, имя, отчество. Заголовок процедуры:

Procedure ReadRuss( Var Stroka: String; Var CodIor: Integer);

где Stroka – вводимая строка; CodIor – код ошибки (номер символа, в котором сделана ошибка).

Номер группы, оценки по предметам должны быть переменными целого типа. Для ввода целого числа можно воспользоваться процедурой ReadInt. Заголовок процедуры:

Procedure ReadInt (Var IVar : Integer; Var CodIOR: Integer);

где IVar – вводимое число, CodIOR – код ошибки.

Для организации интерфейса можно использовать такие подпрограммы:

Menu – вывод горизонтального меню;

frame – построение рамки для выделения элементов, выводимых на экран;

Middle – процедура вывода на экран строки текста в центре экрана;

и другие.

Для использования стандартных процедур поместим их в отдельный модуль, например Unit Dla_vvoda. В модулях ввода и вывода осуществим подключение этого модуля с помощью Uses-фразы. В этом случае можно обращаться к процедурам модуля так, как если бы они находились непосредственно в программе.

Пример использования обращений в фрагментах программы:

```
// Ввод и запись данных по предметам сессии
  Assign(f1, 'Predmet.dat');
  ReWrite(f1);
  For i:= 1 To 3 Do
     Begin
       Write(i,'--'); ReadRuss(sss, code);// название предмета вводится русскими буквами
     Writeln(f1,sss);
     End;
 Close(f1);
// Ввод и запись данных результатов сессии
     Write ('Имя файла'); Readln(fName);
  Assign(f, FName);
  ReWrite(f);
  For i:= 1 To 9 Do
    Begin
     Write(i,'Номер группы---'); ReadInt (b.N_G, k); //Ввод целого числа
     Write(i,'Фамилия---'); ReadRuss(b.name.f, code);//Ввод русскими буквами
     Write(i,'Имя---'); ReadRuss(b.name.name1,code);
```

```
          Write(i,'Отчество---'); ReadRuss(b.name.name2,code);
          For j:= 1 To 3 Do
            Begin
              Write(pr[j],'---'); ReadInt (b.rez[j],k); //Ввод целого числа
            End;
          Write(f,b);
        end;
      Close(f);
```

Просмотрите работу программы. Обратите внимание на подключение стандартных подпрограмм.

***Самостоятельная работа***

*Процедура вывода графического меню:*

```
{=========================Menu================================}
{==============Процедура вывода горизонтального меню===============}
{================Ypos – позиция вывода по вертикали================}
{==================================================================}
procedure menu(Ypos:Byte);
Uses Frame; {-------------- Если подпрограмма находится в модуле Frame----------}
{==========Процедуры для установки цвета фона и текста=============}
procedure b;begin textbackground(blue) end;
procedure ma;begin textbackground(magenta) end;
procedure r;begin textbackground(red) end;
procedure w; begin textcolor(white) end;
procedure y; begin textcolor(yellow) end;
{==================Здесь нужны исправления=========================}
{===========Исправьте название пуктов меню в соответствии с Вашей программой.
Вместо
                                    ВВОД, ЗАПРОС, ВЫХОД,
                запишите нужные Вам названия ========================}
                {==count_I – количество пунктов меню. Исправьте 3 на нужное Вам количество.====}
  const count_i=3;iname:array[1..count_i] of string=
        ('ВВОД','ЗАПРОС','ВЫХОД');
{========В массиве ipos указываются позиции вывода пунктов меню по горизонтали.
Вместо
                                    10, 30, 50
                запишите свои координаты.=========================}
        ipos:array[1..count_i] of byte = (10,30,50);
{=========================+++++==================================}
var
  posK: Byte;                   {Последняя позиция вывода рамки}
  pos: byte;                    {номер текущего пункта меню}
  q:boolean;                    {определяет конец работы с меню}
  i_done:boolean;               {Для определения сделан ли ввод}
  c:char;                       {Для считывания кода клавиши}
begin
  b;clrscr;ma;y;
  gotoxy(ipos[1],Ypos);write(iname[1]);
  b;w;
{=======================Вывод пунктов меню====================}
```

```
    for pos := 2 to count_i do begin gotoxy(ipos[pos],Ypos);
                            write(iname[pos]);
                        end;
{============================Вывод рамки=======================}
   If ipos[1] > 4 Then pos := ipos[1] - 4 Else pos := 1;
   If ipos[count_i] > 70 Then posK := 79 Else posK := ipos[count_i]+10;
   Frame(pos,Ypos-1,PosK,Ypos+1,3,2);
{====================Начальная инициализация====================}
    pos:=1;q:=false;
    repeat
{====Считывание клавиши====}
        c:=readkey;
        case c of
            #13: Begin                          {Клавиша Enter}
                 case pos of
{====================Здесь нужны исправления=====================}
{=============Запишите обращения к подпрограммам=================}
                  1: VvodID(Fname);  {--------Обращение к 1-й подпрограмме----------}
                  2: Zapros(Fname);    {--------Обращение к 1-й подпрограмме----------}
                  3: q:=true;               {--------Выход. Последний пункт меню.------------}
{=============================================================}
                 end;
{----Настройка меню после работы процедуры---------------}
                 b;clrscr;ma;y;
                 gotoxy(ipos[1],Ypos);write(iname[1]);
                 b;w;
                 for pos := 2 to count_i do begin
                                gotoxy(ipos[pos],Ypos);
                                write(iname[pos]);
                                   end;
              If ipos[1] > 4 Then pos := ipos[1] - 4 Else pos := 1;
              If ipos[count_i] > 70 Then posK := 79 Else posK := ipos[count_i]+10;
              Frame(pos,Ypos-1,PosK,Ypos+1,3,2);
                  pos:=1;
{-------------------------------------------------------------------------------------------}
                End;
            #27: q:=true;                       {Клавиша Esc}
            #0: begin
{-------------Необходимо использовать расширенный код--------------------------}
                c:=readkey;
                case c of
                   #75:begin                    {Стрелка влево}
                       gotoxy(ipos[pos],Ypos);write(iname[pos]);
                       if pos =1 then pos:=count_i else dec(pos);
                       ma;y;
                       gotoxy(ipos[pos],Ypos);write(iname[pos]);
                       b;w;
                      end;
```

```
                #77:begin                       {Стрелка вправо}
                    gotoxy(ipos[pos],Ypos);write(iname[pos]);
                    if pos =count_i then pos:=1 else inc(pos);
                    ma;y;
                    gotoxy(ipos[pos],Ypos);write(iname[pos]);
                    b;w;
                   end;
            end; {Case}
{-------------------------------------------------------------------------------------------}
            end;{Расширенный код}
      end; {case}
   until q;          {Цикл считывания клавиш}
While KeyPressed Do Reg:=Readkey;
end;
```

*Вариант 1*

*Условие задачи*

Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота).

Для повышения качества программного комплекса, который Вы разрабатывали на предыдущем занятии, включите в программу стандартные подпрограммы и запишите программу в рабочую директорию.

1. Ознакомьтесь с библиотекой процедур. Отметьте для себя, какие из имеющихся процедур Вы могли бы использовать в своей программе. Осуществите защиту ввода.

2. Ознакомьтесь с текстом программы Menu. Определите, какие изменения Вы должны сделать в программе для подключения ее к своему комплексу. Подключите процедуру в разрабатываемый Вами комплекс.

3. Проверьте работоспособность программы. Если необходимо, воспользуйтесь методами отладки.

*Вариант 2*

*Условие задачи*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату.

Для повышения качества программного комплекса, который Вы разрабатывали на предыдущем занятии, включите в программу стандартные подпрограммы и запишите программу в рабочую директорию.

1. Ознакомьтесь с библиотекой процедур. Отметьте для себя, какие из имеющихся процедур Вы могли бы использовать в своей программе. Осуществите защиту ввода.

2. Ознакомьтесь с текстом программы Menu. Определите, какие изменения Вы должны сделать в программе для подключения ее к своему комплексу. Подключите процедуру в разрабатываемый Вами комплекс.

3. Проверьте работоспособность программы. Если необходимо, воспользуйтесь методами отладки.

*Вариант 3*

*Условие задачи*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте

несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех.

Для повышения качества программного комплекса, который Вы разрабатывали на предыдущем занятии, включите в программу стандартные подпрограммы и запишите программу в рабочую директорию.

1. Ознакомьтесь с библиотекой процедур. Отметьте для себя, какие из имеющихся процедур Вы могли бы использовать в своей программе. Осуществите защиту ввода.

2. Ознакомьтесь с текстом программы Menu. Определите, какие изменения Вы должны сделать в программе для подключения ее к своему комплексу. Подключите процедуру в разрабатываемый Вами комплекс.

3. Проверьте работоспособность программы. Если необходимо, воспользуйтесь методами отладки.

*Вариант 4*

*Условие задачи*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой–участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде–участнице, а также упорядоченный список в соответствии с набранным количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5.

Для повышения качества программного комплекса, который Вы разрабатывали на предыдущем занятии, включите в программу стандартные подпрограммы и запишите программу в рабочую директорию.

1. Ознакомьтесь с библиотекой процедур. Отметьте для себя, какие из имеющихся процедур Вы могли бы использовать в своей программе. Осуществите защиту ввода.

2. Ознакомьтесь с текстом программы Menu. Определите, какие изменения Вы должны сделать в программе для подключения ее к своему комплексу. Подключите процедуру в разрабатываемый Вами комплекс.

3. Проверьте работоспособность программы. Если необходимо, воспользуйтесь методами отладки.

*Вариант 5*

*Условие задачи*

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков.

Для повышения качества программного комплекса, который Вы разрабатывали на предыдущем занятии, включите в программу стандартные подпрограммы и запишите программу в рабочую директорию.

1. Ознакомьтесь с библиотекой процедур. Отметьте для себя, какие из имеющихся процедур Вы могли бы использовать в своей программе. Осуществите защиту ввода.

2. Ознакомьтесь с текстом программы Menu (файл Вывод_меню.rtf). Определите, какие изменения Вы должны сделать в программе для подключения ее к своему комплексу. Подключите процедуру в разрабатываемый Вами комплекс.

3. Проверьте работоспособность программы. Если необходимо, воспользуйтесь методами отладки.

## Лабораторный практикум № 11 Документирование программного продукта. Ввод программного комплекса в эксплуатацию

*Продолжительность: 90 минут.*

*Дисциплина «Технология программирования». Юнита 4.*

Предназначено для обучающихся направления «Информатика и ВТ» в соответствии с учебным планом.

*Цель занятия:* изучить минимальный набор документов для внедрения программы. Изучить способы оформления программного продукта.

*Вводная часть*

*Документация на разработанный программный продукт*

Процесс документирования предусматривает формализованное описание информации, созданной в течение жизненного цикла программного обеспечения. Данный процесс состоит из набора действий, с помощью которых планируют, проектируют, разрабатывают, выпускают, редактируют, распространяют и сопровождают документы, необходимые для всех заинтересованных лиц, таких как руководство, технические специалисты и пользователи системы. Для представления и детализации структуры и содержания технологической документации жизненный цикл программного средства подразделяют:

- системный анализ и проектирование программного средства, разработка и интегрирование программных компонентов;
- тестирование компонентов и комплексов программ;
- испытания программных средств;
- сопровождение и конфигурационное управление версиями программных средств.

Каждый этап снабжается сопроводительной документацией.

До недавнего времени документация выходила в напечатанном виде. Однако сейчас описание программ распространяется, используя электронные средства, что позволяет намного проще и дешевле обновлять документацию. С программным продуктом можно связать справочные файлы, обеспечивая при выполнении задач конспектную подсказку. Кроме того, имеется возможность встроить в продукт мастера и обучающие программы. При написании руководств можно использовать несколько приложений.

Документация на программный продукт, компонент или модуль программного средства должна содержать:

- техническое задание (ТЗ) и/или спецификацию требований на разработку программы;
- описание программы в виде печатного документа;
- фрагмент руководства пользователя – описание применения программного компонента;
- исходный текст программы в виде печатного документа;
- исходный и объектный код программы на магнитных или иных носителях.

Для учебного комплекса техническим заданием можно считать разработанный на первом лабораторном практикуме раздел «Постановка задачи».

*Основные требования к описанию программы*

Описанию программы предшествует ее *спецификация*. В разделе «Спецификация» приводится точное название программы и ее состав. Наименования модулей, их взаимодействие, способы подключения должны быть представлены в виде таблицы:

| Обозначение | Наименование | Примечание |
|---|---|---|
| | | |
| | | |

Графы спецификации заполняют следующим образом:

- в графе «Обозначение» указывают обозначение основных программных компонентов;

- в графе «Наименование» указывают полное наименование соответствующего компонента;
- в графе «Примечание» – дополнительные сведения, относящиеся к записанным в спецификации программам.

Раздел «Описание программы» согласно ГОСТ 19.402–78 должен содержать следующие разделы:

- общие сведения;
- функциональное назначение;
- описание логической структуры;
- используемые технические средства;
- вызов и загрузка;
- входные данные;
- выходные данные.

Отдельные разделы можно объединять. Некоторые пункты этого раздела повторяют разделы технического проекта. Такие повторения предусмотрены ГОСТом, так как на этапе рабочего проекта возникают некоторые дополнения или изменения в составе технических средств или программе. Здесь приводятся более конкретные и точные данные.

В разделе «Общие сведения» должны быть указаны: обозначение и наименование программы; программное обеспечение, необходимое для функционирования программы; языки программирования, на которых написана программа.

В разделе «Функциональное назначение» должны быть указаны классы решаемых задач и (или) назначение программы и сведения о функциональных ограничениях на применение.

В разделе «Описание логической структуры» должны быть указаны: используемые методы; структура программы с описанием функций составных частей и связи между ними; связи программы с другими программами. Описание логической структуры программы выполняют с учетом текста программы на исходном языке.

В разделе «Используемые технические средства» должны быть указаны типы ЭВМ и устройств, которые используются при работе программы.

В разделе «Вызов и загрузка» должны быть указаны способ вызова программы с соответствующего носителя данных, входные точки в программу.

В разделе «Входные данные» должны быть указаны: характер, организация и предварительная подготовка входных данных, формат, описание и способ кодировки входных данных.

В разделе «Выходные данные» должны быть указаны: характер, организация и предварительная подготовка выходных данных, формат, описание и способ кодировки выходных данных.

*Содержание руководства пользователя (описание применения)*

Раздел должен содержать основные характеристики программы, комплектность и сведения об эксплуатации (ограничение применения, минимальная конфигурация технических средств). В разделе описываются:

- условия выполнения программы;
- выполнение программы;
- сообщения оператору.

В разделе «Условия выполнения программы» должны быть указаны условия, необходимые для выполнения программы (минимальный и/или максимальный состав аппаратурных и программных средств и т.п.).

В разделе «Выполнение программы» должна быть указана последовательность действий оператора, обеспечивающих загрузку, запуск, выполнение и завершение программы. В разделе приводятся сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения. Перечисляется порядок и последовательность ввода исходных данных и получения результатов расчета.

В разделе «Сообщения оператору» должны быть приведены тексты сообщений, выдаваемых в ходе выполнения программы, описание их содержания и соответствующие действия оператора.

Содержание разделов допускается иллюстрировать поясняющими примерами, таблицами, схемами.

*Запись на магнитный носитель и поставка программного продукта*

С самого начала работы над программным продуктом необходимо принять решение о том, как распространять систему: на дискетах или CD-ROM либо переслать ее через Internet или intranet, либо просто установить ее на совместно используемом диске.

Кроме того, необходимо обеспечить пользователя информацией о том, как инсталлировать программный продукт. Пользователь составляет свое мнение о системе, начиная с ее установки, поэтому следует сделать инсталляцию максимально простой. Лучше всего создать программу установки (обычно SETUP.EXE) и дать к ней некоторые пояснения (файл README).

*Оформление программного продукта*

Работа любого профессионально разработанного программного комплекса начинается с заставки, являющейся ее визитной карточкой. Заставка может содержать название программы, информацию о ее назначении, фамилию автора программы или название фирмы. Можно вынести в заставку некоторые особенности работы программы.

В заставке могут использоваться различные программные решения, зависящие от того, в каком видеорежиме выводится заставка (графическом или текстовом), от объема информации, содержащейся в заставке, и многих других факторов.

Заставка может состоять из одного, двух или более кадров. В случае использования заставки из нескольких кадров их смена осуществляется по нажатию какой-либо клавиши, кнопки манипулятора мышь или автоматически через определенный промежуток времени. Демонстрация заставки может сопровождаться звуковыми эффектами.

Немаловажным фактором при оценке качества программы является красивый и наглядный вывод. При выводе информации необходимо предусмотреть пояснения результатов. Если чтение информации облегчается, то организовать табличный вывод. Вывод единичного значения лучше представить в центре экрана. Некоторую информацию, имеющую особо важное значение, можно выделить цветом или обвести рамкой.

***Практическая часть***

*Описание программного продукта*

Продолжим работу над учебным примером. Условие задачи:

Разработать программный комплекс по обработке результатов сессии на курсе. Программный комплекс позволяет получать следующие сведения:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

*Спецификация программы*

Программный комплекс содержит следующие модули:

| Обозначение | Наименование | Примечание |
|---|---|---|
| UCHKOML | UCHKOML.pas | Головная программа. Организует обращение к блокам ввода и вывода с помощью горизонтального меню |
| Menu | MenuG.pas | Модуль, содержащий процедуру вывода горизонтального меню |
| Vvod | VvodidG.pas | Модуль, организующий ввод исходной информации. Создает файл «Предметы» и файл «Расписание». Для работы модуля необходимы процедуры, расположенные в модуле Dla_vvoda |
| Dla_vvoda | Dla_vvoda.pas | Содержит процедуры ввода целых чисел и строк кириллицы |
| Zapros | ZaprosG.pas | Головной модуль организации запросов. Считывает информацию из файлов и вызывает горизонтальное меню |
| Menu | MenuZapr.pas | Модуль, содержащий процедуру вывода горизонтального меню, адаптированную под модуль запросов |
| Запросы | Zapros_1_2_3_4.pas | Обрабатывает запросы и организует вывод информации. Для работы требует подключения модуля Dla_vivida и модуля |

| Обозначение | Наименование | Примечание |
|---|---|---|
| | | GraphABC |
| Dla_vivoda | Dla_vivoda.pas | Модуль, содержащий процедуры оформления вывода |
| «Предметы» | PREDMET | Текстовый файл предметов |
| «Расписание» | Проба1.dat | Типизированный файл с результатами сессии |

*Описание программы*

**1. Общие сведения.**

Программа «Деканат» предназначена для демонстрации структурного подхода разработки программного комплекса. Программа может функционировать в среде операционной системы Windows и дополнительного программного обеспечения не требует.

Программа написана на языке PascalABC версии 3.0.1.35, предназначенной специально для обучения программированию. Среда программирования PascalABC относится к классу FreeWare, то есть свободно распространяемому ПО.

**2. Функциональное назначение.**

Функциональное назначение – демонстрация учебного примера. Рассматривается задача обработки результатов сессии. В программе выдаются ответы на четыре запроса: фамилии обучающихся-задолжников, наименование предмета с высшим баллом, процент отличников и медалистов, номера групп в порядке убывания средней успеваемости.

Количество обрабатываемых данных ограничено (20 обучающихся) в связи с учебной целью программы.

3. Логическая структура.

Программа имеет модульную структуру. Состав модулей представлен в спецификации программы. Разработанный комплекс состоит из головной программы и семи модулей. Основная программа осуществляет обращение к модулю ввода (VvodidG.pas) и модулю вывода (ZaprosG.pas), а также выполняет выход из программы.

Модуль ввода (VvodidG.pas) осуществляет запись на диск результатов сессии. Предлагаются два варианта ввода: ввод данных по предметам и ввод результатов сессии. Названия предметов, по которым сдавались предметы, записываются в текстовый файл с названием «PREDMET». Результаты сессии записываются в типизированный файл. Имя файла вводится с клавиатуры. Содержание файлов используется для составления запросов. В целях защиты ввода для работы модуля VvodidG.pas необходимы процедуры, организующие ввод целых чисел и строк кириллицы, расположенные во вспомогательном модуле Dla_vvoda.

Модуль вывода организует обращение к запросам. Реализация вывода по запросам расположена в модуле Zapros_1_2_3_4.pas. В этом модуле используются вспомогательные процедуры красивого вывода (вывод рамки и центрирование вывода), расположенные в модуле Dla_vivoda.pas.

Обращение к пунктам меню осуществляется с помощью подпрограммы меню. Для головной программы – это процедура MenuG. Для запросов – MenuZapr.pas. Логическая структура программы представлена следующей схемой (рисунок 24).

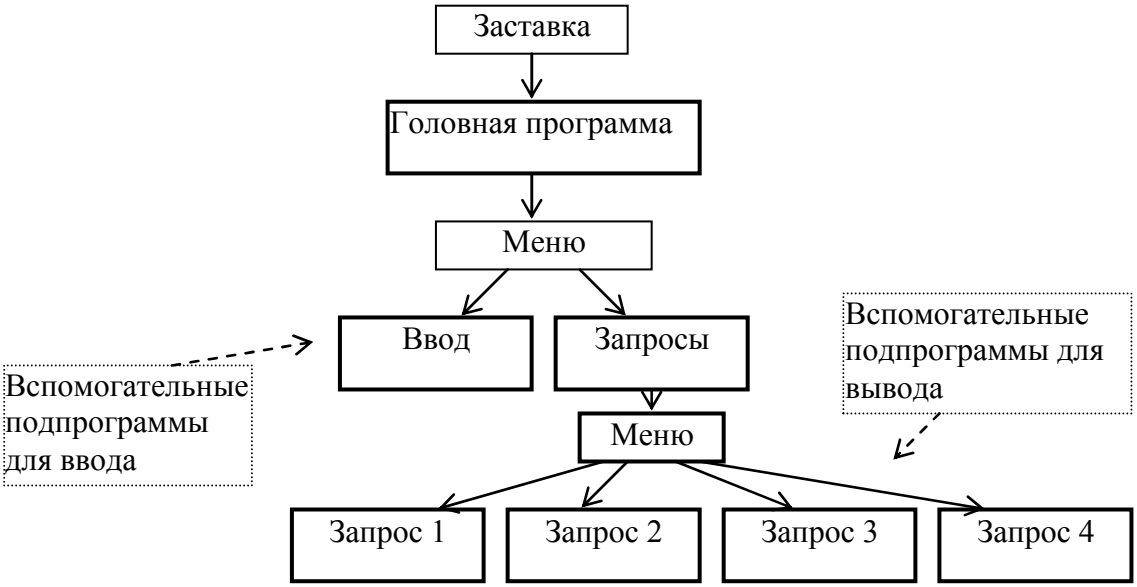



Рисунок 24. Логическая структура программы

4. Используемые технические средства.

Для работы разработанного комплекса используется только базовая конфигурация компьютера. Дополнительных технических средств не требуется.

5. Вызов и загрузка.

Программа вызывается с помощью файла UCHKOML.pas при наличии системы программирования PascalABC или с помощью файла UCHKOML.exe в среде операционной системы Windows. В первом случае все составляющие модули и файлы должны находиться в текущей директории.

6. Входные данные.

Входными данными в этой задаче являются:

- Массив названий предметов, по которым тестировались обучающиеся курса (текстовый файл).
- Информация по каждому обучающемуся курса: (типизированный файл):

a) номер группы;
b) фамилия;
c) имя;
d) отчество;
e) оценки по предметам сессии (массив).

7. Выходные данные:

- фамилии обучающихся, имеющих задолжности хотя бы по одному предмету;
- название предмета, который был сдан лучше всех;
- процент обучающихся, сдавших все экзамены на 5 и 4;
- номера групп в порядке убывания средней успеваемости.

Каждый вывод оформлен в отдельном окне.

***Руководство пользователя***

**1. Условия выполнения программы.**

**В связи с тем, что программа носит учебный характер, дополнительных условий по наличию программных и аппаратных средств не требуется.**

**2. Выполнение программы.**

**Выполнение программы начинается с заставки. Заставка содержит название программы, информацию о ее назначении и разработчиках. Заставка состоит из нескольких кадров, которые меняются динамически без участия оператора.**

**Далее выводится меню, в котором предлагаются три варианта: ввод, запрос, выход. Естественно, прежде чем переходить к запросам, необходимо организовать ввод исходной информации – сформировать файлы данных. Если данные на диске имеются, то можно начинать и с запросов.**

**Информация по предметам:**

- **количество предметов;**
- **название предмета № 1;**
- **название предмета № 2;**
- **название предмета № 3 и т.д.**

**Информация по результатам сессии:**

- номер группы;
- фамилия;
- имя;
- отчество;
- оценки по предметам сессии (массив).

**При выборе пункта меню «Запрос» оператору будет предоставлена возможность выбора одного из четырех видов запросов. После выполнения каждого запроса осуществляется возврат в меню запросов.**

**3. Сообщения оператору.**

Оператор должен внимательно следить за сообщениями программы. При выборе пункта меню «Ввод» последует предупреждение о вводе новой информации. Старая информация при этом будет уничтожена. Ввод данных сопровождается подсказками.

*Оформление программного продукта*

**В учебном комплексе «Деканат» в качестве заставки используются сообщения о представляемом продукте и разработчиках программы. Сообщения расположены в трех кадрах, которые автоматически меняют друг друга.**

**Для организации такой заставки включена процедура Zast_cs. Просмотрите листинг процедуры.**

```
Procedure Zast_cs;
Var ch: Char; {Используется для очистки буфера}
Begin
ShowCursor;
{============================================================}
{==========Здесь можно изменить текст сообщений==============}
{================Цвета сообщений и фона======================}
{=============== Время задержки экрана =====================}
{============================================================}
 ColorStr('Бригада разработчиков № 1',10); //Цветной вывод текста в центре экрана
 ColorStr('представляет',10);              //Цветной вывод текста в центре экрана
 Delay(4000);
 TextBackGround(7);                        // Изменение цвета фона
 ClrScr;
 TextColor(0);                             //Изменение цвета шрифта
 Middle('Подсистему обработки запросов успеваемости',10);
 Middle('в информационной системе "Успеваемость"',11);
```

```
 GoingOut('Продолжение - любая клавиша',6,50,25);
 While KeyPressed do Ch := ReadKey; {Очищаем буфер}
 Showcursor;
End;
```

**В данной процедуре используются подпрограммы:**

| | |
|---|---|
| Goingout | Приостанавливает работу программы и выводит сообщение желаемого цвета и в нужном месте экрана |
| Setnorm | Нормальная форма курсора |
| Setnocur | Невидимый курсор |
| middle | Выводит на экран строку текста, центрируя ее |
| Colrstr | Вывод строки с центрированием и изменением яркости |

**Информация по запросам выводится в рамке и располагается в центре экрана. При выводе информации используются подпрограммы:**

| | |
|---|---|
| goingout | Приостановка работы программы и вывод сообщения желаемого цвета в нужном месте экрана |
| GoingOutColor | Задержки экрана, подсказка о дальнейших действиях пользователя, установка цвета фона и символов |
| frame | Построение рамки для выделения элементов, выводимых на экран |
| Middle | Вывод строки текста с центрированием |
| ColrStr | Вывод строки с центрированием и изменением яркости |

***Самостоятельная работа***

*Вариант 1*

Составьте документацию на разработанный программный продукт.

*Условие задачи*

Разработать программный комплекс, который представляет собой подсистему, обслуживающую учебную часть вуза. Программная подсистема позволяет получать информацию по проведению занятий на первом курсе факультета N: выдает названия предметов, которые ведет преподаватель с фамилией Fam; определяет количество занятий в неделю по каждому из предметов; выводит названия предметов, занятия по которым проводятся в заданный день (понедельник, вторник, среда, четверг, пятница, суббота).

1. Создайте заставку для своего программного продукта. Воспользуйтесь стандартной процедурой Zast_cs. В этой процедуре необходимо сделать изменения в соответствии с Вашей задачей.

2. Продумайте экраны вывода. Для оформления можно воспользоваться стандартными процедурами.

3. Составьте документ из раздела сопроводительных документов «Описание программы». Проследите, чтобы документ содержал все подразделы: спецификация, общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные. Воспользуйтесь редактором OpenOffice.org Write.

4. Составьте документ «Руководство пользователя». Воспользуйтесь редактором OpenOffice.org Write.

Составьте документацию на программный продукт:

- техническое задание скопируйте из документа «Постановка задачи». Если в процессе разработки были внесены изменения в постановку задачи, то соответствующим образом скорректируйте техническое задание;

- добавьте раздел «Описание программы»;
- добавьте документ «Руководство пользователя».

Проверьте наличие сопроводительных документов:

- функциональные диаграммы и диаграммы потоков данных;
- спецификации модулей;
- схемы программ.

*Вариант 2*

Составьте документацию на разработанный программный продукт.

*Условие задачи*

Разработать программный комплекс «Отдел кадров». Подсистема должна обслуживать следующие запросы: выводить список сотрудников, у которых в заданном месяце день рождения; выводить список сотрудников, проживающих на заданной улице; выводить список улиц, на которых проживают сотрудники, в алфавитном порядке; выводить список сотрудников, имеющих минимальную заработную плату.

1. Создайте заставку для своего программного продукта. Воспользуйтесь стандартной процедурой Zast_cs. В этой процедуре необходимо сделать изменения в соответствии с Вашей задачей.
2. Продумайте экраны вывода. Для оформления можно воспользоваться стандартными процедурами.
3. Составьте документ из раздела сопроводительных документов «Описание программы». Проследите, чтобы документ содержал все подразделы: спецификация, общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные. Воспользуйтесь редактором OpenOffice.org Write.
4. Составьте документ «Руководство пользователя». Воспользуйтесь редактором OpenOffice.org Write.

Составьте документацию на программный продукт.

- Техническое задание скопируйте из документа «Постановка задачи». Если в процессе разработки были внесены изменения в постановку задачи, то соответствующим образом скорректируйте техническое задание.
- Добавьте раздел «Описание программы».
- Добавьте документ «Руководство пользователя».

Проверьте наличие сопроводительных документов:

- Функциональные диаграммы и диаграммы потоков данных.
- Спецификации модулей.
- Схемы программ.

*Вариант 3*

Составьте документацию на разработанный программный продукт.

*Условие задачи*

Разработать программный комплекс по обслуживанию риэлтовской компании. Программа должна давать информацию по следующим параметрам: общая площадь, жилая площадь, количество комнат, наличие санузла и его характеристики, наличие лоджии, панельный или кирпичный дом, стоимость квартиры. Сформулируйте несколько критериев, по которым можно отобрать ту или иную квартиру для покупки и, основываясь на этих критериях, выберите сведения о ней. Если подходящих квартир несколько, то выведите сведения обо всех.

1. Создайте заставку для своего программного продукта. Воспользуйтесь стандартной процедурой Zast_cs. В этой процедуре необходимо сделать изменения в соответствии с Вашей задачей.
2. Продумайте экраны вывода. Для оформления можно воспользоваться стандартными процедурами.
3. Составьте документ из раздела сопроводительных документов «Описание программы». Проследите, чтобы документ содержал все подразделы: спецификация, общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные. Воспользуйтесь редактором OpenOffice.org Write.
4. Составьте документ «Руководство пользователя». Воспользуйтесь редактором OpenOffice.org Write.

Составьте документацию на программный продукт.

- Техническое задание скопируйте из документа «Постановка задачи». Если в процессе разработки были внесены изменения в постановку задачи, то соответствующим образом скорректируйте техническое задание.

- Добавьте описание программы.
- Добавьте документ «Руководство пользователя».

Проверьте наличие сопроводительных документов:

- функциональные диаграммы и диаграммы потоков данных;
- спецификации модулей;
- схемы программ.

*Вариант 4*

Составьте документацию на разработанный программный продукт.

*Условие задачи*

Разработать программный комплекс для подведения итогов Олимпийских игр. В программу пользователь должен ввести количество медалей разного достоинства, завоеванное каждой командой-участницей. Программа подсчитывает общее число медалей и соответствующее число очков. Программа должна выдавать информацию по каждой команде-участнице, а также упорядоченный список в соответствии с набранным количеством очков. Количество очков определяется по следующему правилу: за золотую медаль команда получает 7 очков, за серебряную – 6, за бронзовую – 5.

1. Создайте заставку для своего программного продукта. Воспользуйтесь стандартной процедурой Zast_cs. В этой процедуре необходимо сделать изменения в соответствии с Вашей задачей.
2. Продумайте экраны вывода. Для оформления можно воспользоваться стандартными процедурами.
3. Составьте документ из раздела сопроводительных документов «Описание программы». Проследите, чтобы документ содержал все подразделы: спецификация, общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные. Воспользуйтесь редактором OpenOffice.org Write.
4. Составьте документ «Руководство пользователя». Воспользуйтесь редактором OpenOffice.org Write.

Составьте документацию на программный продукт.

- Техническое задание скопируйте из документа «Постановка задачи». Если в процессе разработки были внесены изменения в постановку задачи, то соответствующим образом скорректируйте техническое задание.
- Добавьте описание программы.
- Добавьте документ «Руководство пользователя».

Проверьте наличие сопроводительных документов:

- Функциональные диаграммы и диаграммы потоков данных.
- Спецификации модулей.
- Схемы программ.

*Вариант 5*

Составьте документацию на разработанный программный продукт.

Условие задачи:

Разработать программный модуль для обработки информации о подписных изданиях. Программный комплекс должен обслуживать следующие запросы:

А. По заданному номеру месяца и названию издания найти количество экземпляров, подлежащих доставке.

Б. По фамилии найти список подписных изданий данного подписчика.

В. По заданному названию издания и номеру месяца определите участок, получающий больше всего экземпляров.

Г. По заданному участку доставки и месяцу определите издание, на которое подписалось наибольшее число подписчиков.

1. Создайте заставку для своего программного продукта. Воспользуйтесь стандартной процедурой Zast_cs. В этой процедуре необходимо сделать изменения в соответствии с Вашей задачей.
2. Продумайте экраны вывода. Для оформления можно воспользоваться стандартными процедурами.

3. Составьте документ из раздела сопроводительных документов «Описание программы». Проследите, чтобы документ содержал все подразделы: спецификация, общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные. Воспользуйтесь редактором OpenOffice.org Write.

4. Составьте документ «Руководство пользователя». Воспользуйтесь редактором OpenOffice.org Write.

Составьте документацию на программный продукт.

- Техническое задание скопируйте из документа «Постановка задачи». Если в процессе разработки были внесены изменения в постановку задачи, то соответствующим образом скорректируйте техническое задание.
- Добавьте описание программы.
- Добавьте документ «Руководство пользователя».

Проверьте наличие сопроводительных документов:

- Функциональные диаграммы и диаграммы потоков данных.
- Спецификации модулей.
- Схемы программ.

## ЛИТЕРАТУРА


**Основная учебная**

1. Алексеев, А.А. Основы параллельного программирования с использованием Visual Studio 2010 [Электронный ресурс]: учебное пособие/ Алексеев А.А.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2013.— 138 c.— http://www.iprbookshop.ru/16714.— ЭБС «IPRbooks»
2. Фарафонов А.С. Программирование на языке высокого уровня [Электронный ресурс]: методические указания к проведению лабораторных работ по курсу «Программирование»/ Фарафонов А.С.— Электрон. текстовые данные.— Липецк: Липецкий государственный технический университет, ЭБС АСВ, 2013.— 32 c.— http://www.iprbookshop.ru/22912.— ЭБС «IPRbooks»
3. Мейер Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс]/ Мейер Б.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 285 c.— http://www.iprbookshop.ru/39552.— ЭБС «IPRbooks»

**Дополнительная**

1. Васильев В.Н. Основы программирования на языке С+ [Электронный ресурс]: учебное пособие/ Васильев В.Н.— Электрон. текстовые данные.— Волгоград: Волгоградский институт бизнеса, Вузовское образование, 2010.— 72 c.— http://www.iprbookshop.ru/11341.— ЭБС «IPRbooks»
2. Шевченко П.Н., Методологии и технологии программирования [Электронный ресурс]: рабочий учебник/ Шевченко, П.Н. - 2010. - http://lib.muh.ru
3. Шевченко П.Н. Организация и обеспечение процесса создания программных средств [Электронный ресурс]: рабочий учебник/ Шевченко, П.Н. - 2010. - http://lib.muh.ru
4. Шевченко П.Н., Технологии проектирования программного обеспечения [Электронный ресурс]: рабочий учебник/ Шевченко, П.Н. - 2009. - http://lib.muh.ru
5. Шевченко П.Н., Языки и системы программирования [Электронный ресурс]: рабочий учебник/ Шевченко, П.Н. - 2010. - http://lib.muh.ru

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ**
**ПО ДИСЦИПЛИНЕ «ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ»**
**НАПРАВЛЕНИЕ ПОДГОТОВКИ**
**09.03.01 «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»**

**Квалификация (степень) – бакалавр**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки В.Г. Буцкая

4414.01.01;МУ.01;1

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

# ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «АНГЛИЙСКИЙ ЯЗЫК. БАЗОВЫЙ КУРС ДЛЯ НЕЛИНГВИСТОВ» (КУРС 1)

МОСКВА 2018

Разработано Л.Д. Захаровой, к.фил.н., доц.
Под ред. В.Н. Базылева, д.фил.н., проф.

Рекомендовано Учебно-методическим советом в
качестве методических указаний для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

# ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «АНГЛИЙСКИЙ ЯЗЫК. БАЗОВЫЙ КУРС ДЛЯ НЕЛИНГВИСТОВ»

Методические указания подготовлены для обучающихся в образовательной организации и предназначены для овладения умениями речевого общения, использования и коррекции речи в профессиональных целях по направлению  в рамках дисциплины «Английский язык. Базовый курс для нелингвистов» (курс 1)

**О Г Л А В Л Е Н И Е**

Стр.

## I ВВЕДЕНИЕ

**Цель практических занятий** заключается в формировании коммуникативной компетенции студента, позволяющей вступать в коммуникацию и уметь ориентироваться и реализовывать коммуникативные намерения в основных ситуациях общения (бытовых, социально-культурных, учебно-производственных); вступать в коммуникацию, задавать вопрос и сообщать о факте или событии, выражать намерение, желание, просьбу, пожелание и т.п.; выражать свое отношение к лицу, предмету, факту, событию.

**Задачи практических занятий:**

- формирование навыков и умений устной (монологической и диалогической) и письменной речи на английском языке;
- активизация грамматических навыков;
- совершенствование лексических навыков, связанных с умением использовать лексико-фразеологические средства языка в речи, подбирать синонимические средства языка;
- совершенствование дискурсивных умений, связанных с оценкой типа текста, вариантов речи нормативного и ненормативного характера;
- формирование лингвокультурной компетенции, предполагающей знакомство с речевым этикетом, стереотипами речевого общения в англоязычной культуре;
- формирование стратегической компетенции, включающей в себя речевую активность, устойчивую потребность в общении на английском языке;
- формирование у обучаемых заинтересованности в самообразовательной деятельности для более глубокого и осмысленного усвоения программных положений учебной дисциплины.

**Особенность** данного вида практических занятий заключается в коммуникативной направленности занятий, предполагающего последовательности осуществления практических и познавательных действий. Коммуникативные игры обладают высокой степенью наглядности и позволяют активизировать изучаемый языковой материал в речевых ситуациях, моделирующих и имитирующих реальный процесс общения.

Решение языковой задачи предусматривает формирование или совершенствование речевых навыков в процессе целенаправленного использования заданного языкового материала в речевой деятельности. Коммуникативная задача заключается в обмене информацией между участниками игры в процессе совместной деятельности.

Коммуникативная игра как особый вид занятия состоит из нескольких частей:

1) *вступительная*. Обучаемые знакомятся с темой, целью, порядком проведения занятия, его значимостью для профессиональной деятельности, критериями оценки качества отработки заданий, рекомендациями по использованию учебной литературы. На этом этапе осуществляется постановка конкретной задачи, моделирующей будущую профессиональную деятельность;

2) *подготовительная*. Обучаемые заняты коллективной работой: они продумывают речевые задачи каждой роли, формулируют реплики, обсуждают их уместность, языковую и речевую грамотность, интонационную реализацию;

3) *практическая*. Этап реализации речевых заданий: учащиеся произносят подготовленные диалоги и монологи, импровизируют на основе подготовленного речевого материала;

4) *заключительная*. Подведения итогов и контроля качества усвоения материала и оценки умений работы с текстами и лингвистическими базами на немецком языке: смысловой, лингвистический и стилистический анализ речевого материала. Подводятся итоги занятия, обучаемым выставляются оценки.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 1. ДЕЛОВАЯ ИГРА «ЗНАКОМСТВО», «ПРОСТОЕ ПРЕДЛОЖЕНИЕ»

***Цель работы***: *усвоение речевых формул приветствия, прощания, представления, благодарности.*

Для выполнения практического занятия обучающимся необходимо использовать одноязычные словари английского языка (http://www.babla.ru/), карточки с персональными заданиями.

Игра 1. Каждый студент получает карточку, на которой написано имя-фамилия, профессия и возраст. Студенты знакомятся с образцом и каждый готовит о себе подобный рассказ.

My name is … I am seventeen (sixteen, eighteen). I live in Rostov-on-Don. My address is … My telephone number is…

I have just left school and now I am going to enter the University (Institute). I am going to be a teacher (a lawyer). I like my future profession and I am going to do my best to become a good specialist.

I live with my family. It is large (small, not very large) and very good. We love each other very much and always try to help each other and to spend as much time together as we can. I have a lot offriends too.

I am fond of reading and playing computer games. My favourite sport is football (swimming, tennis, hockey). My friends and I often get together to play different games, go for a walk or to the disco or simply talk.

*2. Игра 2. Диалоги. Студенты делятся на группы по 3, и каждый готовит представление двух студентов на основе предложенных образцов.*

Maria: Hello, I'm Maria. Clara: Hello Maria, I'm Clara. Maria: Pleased to meet you.

Jordi: Excuse me, are you Silvio?

Silvio: Yes, I am. What's your name?

Jordi: My name's Jordi.

Maria: Where are you from? Clara: I'm from Barcelona.

Jordi: What's your email address?

Silvio: It's sitvio77@hotserve.com.

Maria: What's your job?

Clara: I'm a student at the university.

Jordi: Are you married?

Silvio: No, I'm not. I'm single.

Interviewer: Do you eat with the clients?

Jenny: No, I don't. I have lunch with the other reps.

Interviewer: And what do you do in the afternoon?

Jenny: At half past three I go to the hotel pool and help the other reps with games.

Interviewer: Games? Jenny: Yes, we organise all kinds of games and competitions for the clients. It's great fun.

Interviewer: Do you play the games?

Jenny: Oh no, I don't. I'm the referee!

Interviewer: So, what do you do in the evening? Do you have dinner with the clients?

Jenny: Yes, I do.

Interviewer: Where do you go for dinner?

Jenny: I take the clients to a restaurant at quarter to eight and then I take them to a nightclub at about half past ten. Sometimes we have special parties and entertainment.

Interviewer: When do you finish work?

Jenny: Well, I leave the nightclub at about half past one in the morning. So I get home at about quarter to two.

Interviewer: What a busy life!

Jenny: Yes. But I love it!

***

- Martin: Hello, my name is Martin Lngel.
- Michael: Our room is next to your room.
- Martin: Oh really? My name is Michael Lentz. And this is Marie Dinsmore.
- Martin: Nice to meet you.
- Marie: Nice to meet you. You can call me
- Martin: And I am Martin.
- Michael: I am Mike. How are you?
- Martin Quite well. How are you?
- Mike: Just fine. It's such a lovely day. Do you have plans?
- Martin: Yes, my wife and I plan to visit S. Paul's Cathedral.
- Mike: That's a great idea.

- Martin: Here comes my wife. Gisela, this is Mike Lentz. And this is Mari Dinsmore. Marie, Mike, this is Gisela.
- Mike: Nice to meet you, Gisela.
- Gisela: Nice to meet you, Mike. Hello Marie.
- Marie: Hello, Gisela. Sorry, we have to go now. Have a good day!

***

- Marie: Where are you from?
- Martin: We come from a village near Munich, Germany.
- Mike: So, you live just outside the city?
- Martin: Our village is 40 minutes from Munich by train.
- Mike: We are from Yorkshire. It's in the middle of the country.
- Marie: We live in a small village, too. About 1000 people live there.
- Gisela: Our village has only 800 inhabitants, but we love it. It is quiet, and we can see the Alps.
- Martin: Are you here on holiday?
- Mike: No, I'm here on business.
- Gisela: We're here for five days. And you?
- Mike: We're here for a week. I have business to take care of, and then we are doing some sightseeing.
- Martin: Perhaps we could have dinner together one night.
- Mike: That would be nice. Enjoy London.
- All: Goodbye!

*Игра 3. Студенты делятся на группы, каждая группа получает карточку со словами. Из этих слов студенты должны составить как можно больше предложений на английском языке. Предложения из одних и тех же слов, но с различным порядком слов засчитываются как разные. Затем каждая из групп презентует свои предложения, а задача других команд – составить предложения со словами другой команды, такие, которые отсутствуют в презентации. Побеждает та команда, которая предложила как можно большее количество предложений и со словами которой конкуренты составили наименьшее количество предложений*

**Карточки со словами**

**№ 1**

| |
|---|
| to eat |
| tasty dish |
| to drink |
| a bread |
| an egg |
| a soup |
| sour |
| sweet |
| warm |

**№ 2**

| |
|---|
| cook food |
| like |
| believe |
| chicken |
| a meat |
| a fish |
| old |
| hard |
| bitter |

**№ 3**

| |
|---|
| a plate |
| a knife |
| a spoon |
| dry |
| salt |
| spicy |
| bring (brought) |
| to do shopping |

**№ 4**

| |
|---|
| a fork |
| a glass |
| a bottle |
| in the afternoon |
| in the evening |
| sometimes |
| to clean |
| to wash up |

**№5**

| |
|---|
| a wine |
| a beer |
| a juice |
| enough |
| satisfied |
| hungry |
| need |
| to take |
| to pay |

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 2. КОММУННИКАТИВНАЯ ИГРА «МОЙ ДОМ», «ЧИСЛО СУЩЕСТВИТЕЛЬНОГО»

***Цель работы***: *усвоение речевых формул, лексики по темам «Моя семья», «Мой дом»; формирование умений монологического описания, а также обиходно-бытового диалога; совершенствование грамматических умений использования имени существительного.*

Для выполнения практического занятия обучающимся необходимы тексты и словари для анализа (http://www.babla.ru/), фотография (http://www.predmet-photo.ru/images/blog/kartinki-s-dnem-rozgdenia/otkritki/s-dnem-rozgdenia-otkritka-021.jpg.

*Игра 1. Каждый студент готовит схематический план своей квартиры, одновременно студент готовит вопросы к собеседнику о его квартире. Далее студенты разбиваются на группы, один отвечает на вопросы о квартире (доме), другие – задают вопросы, они пользуются и приведенными ниже текстами и диалогами.*

We have a nice flat in a new block of flats. Our flat is on the fifth floor of a nine-storied building. It has all modem conveniences: central heating, running hot and cold water, electricity, gas, a lift and a chute to carry rubbish down.

We have a three-room flat which consists of a living-room, a bedroom, a study (which is also my room), a kitchen, a bath-room and a toilet. There are also two closets in our flat. Our flat has two balconies.

The living-room is the largest and most comfortable one in the flat. In the middle of the room we have a square dinner-table with six chairs round it. To the left of the dinner-table there is a wall-unit which has several sections: a sideboard, a wardrobe and some shelves. At the opposite wall there is a piano and a piano stool before it. To the right there is a little table with colour TV set on it. Opposite the TV set there are two cozy armchairs. A divan-bed and a standard lamp are in the left-hand corner. In front of the armchairs there is a small round table for newspapers and magazines. There is a thick carpet on the floor. Two water-colours hung on the wall above the divan-bed. In the evening we usually draw the curtains across the windows, and a red lampshade gives a warm colour to the room.

The bedroom is smaller than the living-room and not so light as there is only one window in it. In this room there are two beds, two dressing-tables and a wardrobe. In the corner of the bedroom there is a small colors TV set. On the dressing table there is an alarm-clock and small lamp with green lamp-shade.

Our study is the smallest room in the flat, but in spice of it, it is very cozy. There isn't much furniture in it, but there are a lot of shelves full of books. It has a writing table, an armchair and a bookcase too. A small round table with cassette-recorder is standing in the right-hand corner of the study. Besides there is a small sofa near the wall opposite the bookcase. This room was my father's study, but as I grew older, it has become my room. And in my opinion it is the best room in our flat. My friends used to come to my place to have a chat or to play chess in the evening, and they say my room is very comfortable. I share their opinion.

***

Amanda: So, where do you live, Pete? Have you got your own house?

Pete: No, I haven't. I've got a modem studio apartment in the centre of town.

Amanda: Has it got a garden?

Pete: No, it hasn't got a garden, but it's got a small terrace.

Amanda: Is there a kitchen in the apartment?

Pete: No, there isn't but there's a kitchen area with a fridge, a cooker and a sink. But I haven't got a microwave.

Amanda: What about furniture?

Pete: I've got a coffee table, and there are two chairs. And I've got a beautiful sofa, I love that sofa,I use it all the time - I eat my meals there because I haven't got a dining table!

Amanda: Is there a TV?

Pete: Yes, of course. And I've got a music system.

Amanda: Have you got a computer?

Pete: Yes, I've got a laptop computer -I use the Internet a lot.

Amanda: And have you got a mobile phone?

Pete: Yes, I have.

*Игра 2. Индивидуальная игра-эстафета. Перед студентами предложения. Каждый по очереди определяет число существительного и образуют (если это возможно) форму другого числа.*

The Oxford Advancer Learner's Dictionary of Current English by Hornby gives us the following definition of the notion "art". ""Art" is the creation or expression of what is beautiful, especially in visual form. Drawing, painting, sculpture, architecture, literature, music, ballet belong to the fine art".

Really when something is extremely beautiful or has great cultural value, we say: "It's art". Art has always been occupation for the few, but has been admired by many. Art reflects feelings and emotions, brings delight and admiration, and makes life pure as it wakens our best hidden qualities. Speaking about arts, we connect this notion with culture. According to the dictionary culture of a community or nation includes all the arts, beliefs and social institutions characteristic of a community or nation. We can speak about either material, or spiritual culture. Art is both.

Russia is a country that can rightfully boast its artistic and cultural traditions, its art galleries attract huge crowds of tourists from all over the world. St. Petersburg is a precious stone in the crown of Russian cities. The Hermitage is famous all over the world for its valuable rare collections of canvases and other art objects covering a span of about seven hundreds years and comprising masterpieces of by Leonardo da Vinci, Titian, Raphael, Rembrandt, and Rubens. The collections illustrate the art of Italy, Spain, Holland, Germany, France, Britain, and Sweden. The West - European Department also includes a fine collection of European Sculpture. People come to admire the collections of tapestry, precious textiles, weapons, ivory, pottery, porcelain and furniture as well.

The Tretyakov Gallery in Moscow, the Russian Museum should be mentioned by all means. This picture gallery was founded by a Russian merchant and a connoisseur of art Pavel Tretyakov in the 19th century. He was especially fond of the works of Peredvizhniki or Wanders- the artists who belonged to the Society of Travelling Art Exhibitions such as Kramskoy, Perov, Ghe and other great Russian painters. The Tretyakov Gallery reflects the whole history of Russian Art. It has a rich collection of early Russian painting including famous icon. The world famous "The Trinity" by Andrey Rublev is exhibited in the gallery.

Speaking about art one should not forget about music, especially classic music. Outstanding Russian composers make the whole world admire their music. One can find a man, who does not know Pyotr Ilyich Tchaicovsky, Michail Glinka, Nikolai Rimsky-Korsakov - the prominent composers of 19th century, and Sergey Rachmaninov, Sergey Prokofiev and Dmitriy Shostakovich.

It was Glinka (1804-1857) who laid the foundation for modern Russian music; his music expressed the temperament of Russian people. His two best known operas "Ivan Susanin" and "Ruslan and Ludmila" were based on Russian folklore and historical legends.

The most famous ballets "Swan Lake", "The Sleeping Beauty", "The Nutcracker" and not less famous operas "The Queen of Spades", "Eugene Onegin" are still excellently staged and performed not only in Russian but in many greatest theatres in the world.

Russia is world famous for its literature. The "golden age" of Russian literature began in the 19th century when such outstanding masters of letters such as Alexander Pushkin, Lermontov, Gogol, Turgenev, and Dostoyevsky created their immortal masterpieces.

Alexander Pushkin, the father of Russian Literature was the authors of more than 700 lyrical poems. He wrote also the volumes of dramatic works, short stories, made adaptations of Russian fairy-tales.

Russia is famous for its architecture. The real jewel of architecture is the Moscow Kremlin with its cathedrals, towers and red brick walls. Just outside the Kremlin walls stands St. Basil's Cathedral, one of the world most astonishing buildings with 8 domes of different designs and colors.

St. Petersburg has great number of real masterpieces of architecture of different styles and is definitely worth visiting and being admired.

Russia is rich also in young talents, new Russian culture is forming. It will appear on the basis of the old one, but its essence will be new. We can hear new voices in music and poetry, new canvases of modern artists, great actors and film directors.

All of them will make their contribution into Russian Culture and Art.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 3. КОММУНИКАТИВНАЯ ИГРА «ГЛАГОЛ», «РАСПОРЯДОК ДНЯ»

***Цель работы***: *усвоение речевых формул, лексики по теме «Распорядок дня»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на группы (команды), в каждой не менее 3-4 и не более 6 человек. Каждая группа получает задание подготовить презентацию. Преподаватель ведет игру. Он называет глаголы.*

*Каждая команда за отведенные 3 минуты должна составить несколько предложений с этим глаголом. Побеждает та команда, которая составила максимальное количество правильных предложений.*

1) to stay; 2) to want; 3) to begin; 4) to fly; 5) to sell; 6) to cut; 7) to add; 8) to lock; 9) to admit; 10) to admit; 11) to watch; 12) to fulfil; 13) to whitewash; 14) to sightsee; 15) to magnify; 16) to strengthen; 17) to foresee; 18) to go out; 19) to go by) 20) take in; 21) to bring about; 22) to take care.

*Игра 2. Каждый студент готовит рассказ о своем распорядке дня, одновременно студент готовит вопросы к собеседнику о его обычном дне. Далее студенты разбиваются на группы, один отвечает на вопросы о распорядке дня, другие – задают вопросы, они пользуются и приведенными ниже диалогами и текстами.*

As many people I have different timetables on weekdays and weekends. I think that it is very important to go to bed before midnight and to get up quite early in the morning, especially on weekdays. Thus you can manage to do everything you plan to do. My major duty is to study at the university. My classes usually begin at about 10 o'clock.

Every day I get up at half past six. I take a cool shower then go jogging to the park near my place. So I do jogging for about thirty minutes and then do some exercises. After this I do not feel sleepy at all. I feel refreshed and full of energy. Besides fresh air and birds singing improves my mood greatly even on Mondays.

I return home at about 8 o'clock, take a shower and then have breakfast. I know the first thing many people do when they get up in the morning is to turn on TV. They do it automatically because they are used to all this artificial noise. And when they have breakfast they watch news or morning programs. Well I think all this is the key to our morning depression or bad mood. It is better to read or hear news later at work. I even hate reading newspapers and entertaining magazines in the morning. I like to talk with my parents while having breakfast.

After breakfast I put all the necessary books into my bag and get dressed. I leave home at about 9 o'clock. I get to my university by bicycle. My classes usually finish at 4 p.m. When classes are over I go to the sports center where I have karate lessons. I have karate lessons three times a week. When I do not have them, I take guitar lessons.

I come home at 7 o'clock. I have dinner and start doing my homework at 8 o'clock. If I have some spare time after doing homework, I play the guitar or read classical literature. I go to bed at about half past eleven. On weekends I usually get up at 8 o'clock and do the same things I do on weekdays except going to university. At 12 o'clock I rehearse with in the rock band where I play the guitar. Our rehearsal ends in the evening.

On weekends I can stay at home and read books, or go somewhere with my friends. I think that it is very useful when one has a timetable to follow. I am sure that this prevents us from wasting precious time.

***

As soon as I wake up I open the bedroom window and breathe in some fresh air. Then I go to have a shower. I start with a warm shower and then I turn the water onto cold and for a few seconds I have a really cold shower and let the water run over me. It really wakes you up.

After that I do a few exercises. I think it's really important to do this, because it makes your body feel good and keep the muscles firm. I usually exercise my stomach muscles and my leg muscles. Then I go and have my breakfast.

I really believe that it's important to have a really good breakfast. I don't think you should just have a cup of tea, like most of the girls do. Anyway, I have orange juice, an egg and some biscuits. After breakfast I go to work.

I work as a model, so I like my job, because it's very interesting and I travel a lot. I usually go to work by taxi. It's starts at about 10 o'clock. I work for about 5 hours with the photographer and he takes a lot of pictures. Such pictures are used in women's weekly magazines.

I finish work about four, so you can see that I only work from about 10 till 4 and then I go home. At home I have a bath, change my clothes and watch TV. Every night I usually go out. I can't cook very well, so I like to eat out.

Well, you can see that I have a good life and I have a good wages when you think about the number of hours I work.

***

On week days I usually get up nearly six o'clock. I do not like to get up early, but I have to, because I have a lot of work to do during the day.

I make my bed, wash my face, put my clothes on and go to the kitchen to have breakfast. My mother usually prepares, breakfast for me, but sometimes I do it myself. If I prepare my breakfast for my own, I should have to get up earlier. I do not like big breakfasts; I prefer a cup of coffee and a sandwich.

Then I go to school. It is rather far from my house and I go there by bus. I have classes till two or three o'clock, it depends on a week day. Then I come home and have dinner. I like a big dinner, because when I come home from school I am hungry.

After my dinner, I have a rest for a couple of hours and then I do my homework. If I have some spare time I do some work about the house. I sweep the floor, dust the furniture and clean the carpets with the vacuum-cleaner. Sometimes my mother asks me to go shopping.

Then I have free time. I go for a walk with my friends or watch TV, or read books or play my favourite computer games. Then I have supper with my family. I like evenings very much, all members of our family get together after work and study and have the opportunity to talk and to discuss our family affairs.

I usually go to bed at about ten o'clock, sometimes at eleven o'clock.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 4. КОММУНИКАТИВНАЯ ИГРА «МОДАЛЬНЫЕ ГЛАГОЛЫ», «ДЕНЬ РОЖДЕНИЯ»

***Цель работы***: *усвоение речевых формул, лексики по теме «День рождения», «Вечеринка»; речевых формул поздравления и пожелания; формирование умений монологического описания, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты и словари для анализа (http://www.babla.ru/), фотография (http://www.predmet-photo.ru/images/blog/kartinki-s-dnem-rozgdenia/otkritki/s-dnem-rozgdenia-otkritka-021.jpg).

*Игра 1. Викторина. Задача студентов вставить пропущенный модальный глагол в предложение. Ответивший правильно получает фишку, победителем становится тот, у кого много фишек.*

1. He … (can't/couldn't) open the window as it was stuck.
2. Interpreters … (may/must) translate without dictionaries.
3. … (Can/May) I use me your bike for today?
4. … (May/Could) you give me the recipe for this cake?
5. I hardly ever see Jane, she … (may/might) have moved to Africa.
6. Take an umbrella. It … (may/can) rain.
7. You … (could/should) stop smoking. You know you … (cannot/must not) buy health.
8. You … (may/must) finish the article as soon as possible.
9. Liz doesn't … (ought to/have to) keep to a diet anymore.
10. Lara … (can/might) get a playstation for her birthday.
11. You … (must not/needn't) read in the dark.
12. My grandfather is retired, so he … (shouldn't/doesn't have to) go to work.
13. The fridge is full, so we … (must not/needn't) go shopping.
14. Our employees … (can/must) sign this agreement.
15. We … (may/ought to) reserve a table in advance if we want to have dinner there.
16. I … (can't/needn't) believe it! You … (have to/must) be joking.
17. Ann … (must/is to) finish school next year.
18. Sorry, I'm late. I … (needed to/had to) wait for the plumber.
19. What time do we … (should/have to) be at the railway station?
20. Don't wait for me tonight. I … (might/must) be late.
21. I … (maynot/can't) watch this film. It's too boring.
22. We've got a dishwasher, so you … (couldn't/needn't) wash-up.

23. You look very pale, I think you … (need/should) stay at home.
24. … (Could/Might) you, please, pass me the mustard?

*Игра 2. Каждый студент готовит сообщение о своем дне рождения, одновременно студент готовит вопросы к собеседнику. Далее студенты разбиваются на группы, один отвечает на вопросы, другие – задают вопросы, они пользуются и приведенными ниже диалогами.*

Birthday is a very wonderful day. Everybody likes to celebrate it. It is a good opportunity to spend time with friends, parents, relatives.

I was born on the 10th of January. In the morning on my birthday my parents lay the presents near my bed. So the first thing I see when I open my eyes is my presents. My Mom and Daddy and my little brother come to my room to congratulate me and to sing "Happy Birthday".

Usually we hold my birthday party in the evening. Once we went to a cafe to celebrate my birthday, but usually we celebrate it at home. We clean the house the day before birthday. In the morning of birthday party day my father goes shopping and buys everything we need.

My mother bakes a cake or pie. By the evening food is cooked, the table is laid. We put on evening suits and dresses and wait for the guests. The flat looks nice and cosy. I am always very glad to meet my guests. I like to get flowers and presents. Mom gives me the telegram from my aunt.

We have an abundant dinner on this day. Mom brings in the birthday cake. I blow the candles out. We dance and sing songs, play games and laugh, joke, tell funny stories. I think that my birthday is one of the best days in a year.

***

When we have time for leisure, we usually need something that can interest and amuse us. There are several ways to do this. In big cities it's often difficult to decide where to go in the evening.

If we want to go out there are a lot of theatres, cinemas and clubs in our country where we can spend our free time. (But in small towns and villages they have no actors of their own. So they invite a group of actors from a big town to show plays.)

People who are fond of music join a musical section where they are taught to play different instruments. Those who like to dance join a dancing section.

People who are interested in sports can join sport sections such as tennis, basket-boll, chess and others. And, of course, all the people use radio or television. They switch on the radio set or TV set and choose the programme they like best of all. People who are interested in sports listen to or watch football and basket-ball matches. Everyone likes to see skating and dancing on the ice.

Some people like music. They listen to concerts of modern and old music, new and old songs and see dances. Television helps us to "visit" different lands, see fish and insects, lakes, rivers and seas. We are shown different countries, cities and people who live there. On TV people could even see both sides of the Moon.

Radio and television extend our knowledge about the world. All that we can do at home. So I think, that ways in which leisure time can be spent are different and interesting!

***

I think celebrating birthdays in flats rather boring. In Russia eating and drinking occupies the most part of a birthday celebrating. Everybody at the table very quickly becomes full and begins to feel sleepy. I hate such birthdays. But there is a good way out if your birthday is in summer, early autumn or late spring. You can have a birthday party in the shape of a picnic. It is very convenient when you have a dacha or a country house. Thus you may not hurry to finish picnicking and stay outside late at night and then go to sleep. And your birthday picnic can have a nice continuation of a fireside friendly night chat. This is also not bad if you do not have an opportunity to celebrate your birthday on a dacha. In this case you choose any place in a forest or a park and have a picnic there. The only thing that may prevent you from enjoying yourselves is weather and mosquitoes. If it rains it is very difficult to make a fire. Besides, is there much pleasure in being wet in the forest? But it is equally unpleasant if it is hot. In hot weather one does not have much desire

to stay by the fire and drink anything alcoholic. And almost in any weather there are armies of mosquitoes wishing to feast on our flesh and blood. So you have to be equipped with effective insect repellents. There are obvious advantages in having birthday picnics. First, it is much cheaper, than celebrating at home or a restaurant. Second, the atmosphere of a picnic is rather special and very friendly. I like it, when several people take part in making a fire and cooking. It is so nice to have a relaxed chat with your friends while sitting by the fire, listening to the sound of nature and, of course, eating tasty food. I am sure, that celebrating a birthday on a picnic, gives more pleasure, than doing it somewhere in the town. It is very important not to take with you any music players or radio sets! Living in cities we got used to hearing these artificial sounds since early morning. I am sure there is no need for doing it in the forest. Singing of birds, cracking of fire burning, wind howling – are the sweetest sounds that we hear so seldom. After a picnic it is a must to collect all the garbage, put it into plastic bags and take it to the nearest garbage bin. And of course one has to be absolutely sure that the fire is put out. The only minus of birthday picnics is that it is rather difficult to have tea with a cake in the forest. So you can invite your guests to your place where you can enjoy it for some time. Picnic is my favorite way of celebrating my birthdays. Sometimes when the weather is rainy I have to celebrate my birthday at home.

*Игра 3. Каждый студент готовит письменное поздравление с днем рождения для: а) своего друга; б) одного из своих родителей; в) одного из своих близких родственников. Задание каждый получает в результате жеребьевки, затем каждое предложение зачитывается вслух.*

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 5. РОЛЕВАЯ ИГРА «В МАГАЗИНЕ», «ИНФИНИТИВНЫЕ КОНСТРУКЦИИ»

***Цель работы***: *усвоение речевых формул, лексики по теме: «магазин», «продукты», «одежда»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/)

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание: «Купить продукты для праздника», «Купить продукты на неделю», «Купить подарок другу на день рождение», «Купить новую одежду (обувь)», «Купить мебель (бытовую технику)», «Купить книгу» и под. Далее каждая группа представляет свой диалог.*

**I**

- When we want to buy something we go to a shop. There are many kinds of shops in every town or city, buy most of them have a food supermarket, a department store, men's and women's clothing stores, grocery, a bakery and a butchery. I like to do my shopping at big department stores and supermarkets. They sell various goods under one roof and this is very convenient. A department store, for example, true to its name, is composed of many departments: ready-made clothes, fabrics, shoes, sports goods, toys, china and glass, electric appliances, cosmetics, linen, curtains, cameras, records, etc. You can buy everything you like there. There are also escalators in big stores which take customers to different floors. The things for sale are on the counters so, that they can be easily seen. In the women" clothing department you can find dresses, costumes, blouses, skirts, coats, beautiful underwear and many other things. In the men's clothing department you can choose suits, trousers, overcoats, ties, etc. In the knitwear department one can buy sweaters, cardigans, short-sleeved and long-sleeved pullovers, woolen jackets. In the perfumery they sell face cream and powder, lipstick, lotions and shampoos. In a food supermarket we can also buy many different things at once: sausages, fish, sugar, macaroni, flour, cereals, tea. At the butcher's there is a wide choice of meat and poultry. At the bakery you buy brown and white bread, rolls, biscuits. Another shop we frequently go to is the greengrocery which is stocked by cabbage, potatoes, onions, cucumbers, carrots, beetroots, green peas and what not. Everything is sold here ready-weighed and packed. If you call round at a dairy you can buy milk, cream, cheese, butter and many other

products. The methods of shopping may vary. It may be a self-service shop where the customer goes from counter to counter selecting and putting into a basket what he wishes to buy. Then he takes the basket to the check-out counter, where the prices of the purchases are added up. If its not a self-service shop, and most small shops are not, the shop-assistant helps the customer in finding what he wants. You pay money to the cashier and he gives you back the change. But there is a very good service called Postal Market. It really helps you to save you time and get goods of high quality. You have just to look through a catalogue, choose the things you like, order them and wait a little to get them.

***

**II**

A: Could you help me, please?

B: Yes. What can I do for you?

A: Can you show me this cellphone, please?

B: Yes, of course. Here you are.

A: How much is it?

A: It's 350 $. But today we have a special offer. These phones are 20 percent off.

B: Oh, great. And do you have it in white?

A: Let me see…yes, here it is.

B: OK, I'll take it. Do you take credit cards?

B: Yes, of course. Can you show me some ID, please?

A: Here is my driving license. Is it OK?

A: Yes. Here is your phone and your check. Thank you and come again, Sir/Madam!

B: Good bye.

A: Good bye.

***

**III**

A: Excuse me, I'd like to try this t-shirt on. Where is the fitting-room?

B: It's down there, on the left.

A: Thank you.

(after some time)

A: Could you give me a smaller size?

B: Sorry, we don't have smaller sizes.

A: Oh, what a pity! And this t-shirt doesn't fit. I'll take another one then.

***

**IV**

A: Hello, I'm sorry, but this cheese I've bought in your shop is out of date and spoiled!

B: Oh, dear. Let me take a look. We are very sorry. You can get a refund or exchange it to a fresh one.

A: Thank you. I'll take another one.

B: Please choose it here. We apologize for the inconvenience.

- Customer: Good morning! Excuse me, I bought this car audio here a day before yesterday and I am not satisfied with the sound. I would like to get my money back, please.
- Salesman: I am very sorry, but I`am afraid we don`t give refunds for car audios. Could you show me your receipt, please?
- Customer: Here you are!
- Salesman: Thank you. Well, you can exchange it for another model of equal value. If the chosen model is more expensive, you can pay the remaining sum.
- Customer: All right. But I need your recommendations.

- Salesman: You are really lucky because we start a sale today. This week we offer Toshiba car audios at bargain rates. Would you like to look at displayed goods?
- Customer: Sure!
- Salesman: These models are $ 50–150 cheaper than usually now. These car audios are 20 – 45% off their normal retail price. They are a great deal. You can compare their technical characteristics. And I would like to turn your attention to the guarantee period: they all have a three year guarantee.
- Customer: I like the design and the technical characteristics of this model.
- Salesman: Today you are very lucky again. It is the last one.
- Customer: All right, I will take it. How much extra should I pay for this car audio?
- Salesman: Just twenty-three pounds.
- Customer: Can I pay by my credit card or do you prefer cash?
- Salesman: Of your choice.
- Customer: Then I will pay in cash.
- Salesman: Fine. Here is your car audio.

*Игра 2. Студенты получают карточки, на каждой слова и выражения. Задача – продолжить предложение, используя инфинитивную конструкцию.*

| | |
|---|---|
| to want him 1/ to make an appointment | to feel us 3/ apologize to somebody |
| to expect her 2/ to arrange an outing | to watch her 4/ train for |
| should like them 3/ to be settled | to notice Pete 5/ refuse doing something |
| would like us 4/ to clear up | to find 1/ one's reason |
| to see you 1/ complain of something/ somebody | to ask me 1/ to stay in the shade |
| to hear them 2/ put up a notice | to tell him 2/ to produce an explanation |

*Игра 3. Студенты получают карточки, на них сложное предложение. Задача – продолжить предложение, используя инфинитивную конструкцию.*

1. Some people remember that Tsiolkovsky had taught them aerodynamics.
2. The professor expects that some of his students will make reports at the studens' conference.
3. We know that «Tragic America» by Th. Dreiser gave a true picture of American capitalist society.
4. The ancient people believed that the sun was moving round the earth.
5. Some scientists consider that Mars is covered with vegetation.
6. Historians suppose that the name «London» had come from two Celtic words.
7. Historians supposed the name «London» to have come from two Celtic words.
8. The professor expects some of his students to make reports at the students' conference.
9. Some people remember Tsiolkovsky to have taught them aero dynamics.
10. Some scientists consider Mars to be covered with vegetation.

11. We know «Tragic America» by Th. Dreiser to have given a true picture of American capitalist society.
12. The ancient people believed the sun to be moving round the earth.
13. The old farmer knew that his daughter was in love with a poor cowboy.
14. He knew that his foreman had fallen in love with Santa.
15. The old man didn't consider that Webb was a suitable husband for his pretty daughter.
16. He believed that Webb was too poor to marry Santa.
17. He expected that she would marry a rich farmer
18. Webb knew that Santa's father was against their marriage.
19. Santa expected that Webb would wait until they could be married

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6. РОЛЕВАЯ ИГРА «НА ПРИЕМЕ У ВРАЧА», «СЛОЖНОЕ ПРЕДЛОЖЕНИЕ»

***Цель работы***: *усвоение речевых формул, лексики по теме: «медицина», «здоровье»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание: «Посетить дантиста», «Посетить терапевта», «Выписать лекарство от кашля» и под. Далее каждая группа представляет свой диалог, они пользуются и приведенными ниже диалогами и текстами.*

Do you know the famous phrase "We are what we eat"? The saying is as old as the hills and means that to be fit and healthy you need to eat proper food.

Do you think carefully about the food you eat or you just don't care? Your answer will fully determine your health condition. Nowadays we have a generally accepted pearl of beauty which is foisted on us by the fashion industry. A beautiful girl is suggested to be very slim and even skinny, tall, long legged and long armed. Many girls do their best to look like top-models whom they see every day on advertisement hoardings and on TV But this glossy beauty, which in most cases is made in special computer programmes, conceals a huge problem, sometimes a mortal problem.

Compared with adults, children need more nutrients, as bones, muscles and blood system in their bodies are developing. These nutrients: carbohydrates, vitamins and minerals, protein and fat provide us with energy necessary for growth, tissue repair, immunity and metabolism. Nowadays there is a tendency among teenagers to follow different diets in order to be in line with their idols. Unfortunately, a passion for diets may turn out a real tragedy. It is very important to keep in mind that you mustn't go on a diet without consulting a dietician beforehand; otherwise it may be really dangerous for your health.

There are girls who try to keep a healthy diet, it may be vegetarian, dairy product or rice diet. However, there are also girls who are sure that the less they eat the healthier they are. And that's a great mistake. When preoccupation with being thin takes over your eating habits, thoughts, and life, it's a sign of a psychic disorder. An eating disorder is an illness which causes deep concern about your everyday diet. Eating disorders frequently appear during the teen years. One of the well-known types of eating disorder is anorexia.

When a person has anorexia, the desire to lose weight becomes more important than anything else. He may even lose the ability to see himself from the side. Because of a person's dread of growing fat or disgust at the sight of his body, the eating process may be very stressful. Thoughts about dieting, food, and body may take up most of the day. There is no more time for friends, family, and other activities he or she used to enjoy. But no matter how thin a person grew, it's never enough. People suffering from anorexia never acknowledge the illness, but it can damage their health and even threaten their life.

So, what is the difference between healthy dieting and anorexia? First of all, healthy dieting is an attempt to control only weight, but anorexia is an attempt to control the whole life and emotions. Secondly, when dieting a

person's self-esteem is based on good mood and improving his or her appearance; as for anorexia, it is based entirely on the fact how many kilogrammes you've managed to lose. Thirdly, the aim of losing weight while dieting is to improve a person's health and appearance, whereas for people having anorexia losing weight is the way to achieve happiness.

Anorexia is a very complicated disease and very difficult to cure as it involves not only body but also one's mind. To recover from anorexia a person has to realize that he has a problem. Only in this case it is possible to get over it. Besides anorexia, there are similar diseases caused by lack of eating, for example, bulimia.

There is also another illness connected with eating disorder – obesity. A person with such a problem loses control over his or her eating. The notion "obesity" is different from "being overweight", though both terms mean that a person's weight is greater than the standard corresponding to his or her height. Obesity occurs when a person eats more calories than he or she uses. Being obese increases the risk of diabetes, heart diseases, arthritis and some kinds of cancer. If one is obese, losing even 5 to 10 per cent of one's weight can delay or prevent some of these diseases.

As you probably know, this problem is quite widespread in the USA. Over the last several decades obesity rates have increased for all population groups in the United States. Approximately nine million children over six years of age are considered obese. From 1980 up to 2008, the prevalence of obesity among children aged 6 to 11 years old tripled from 6.5 per cent to 19.6 per cent. It happens not only because Americans are fond of junk food and eat nothing but hamburgers. The main problem comes from genetics. American people try to cope with obesity by banning sodas, junk food and candy at school districts.

I hope that having read this topic, you have learned something useful for yourself. Nothing can be more important in the world than your health. It means that your body is your temple, which should be taken proper care of. I can give you an example of such care. A sensible, well-balanced diet will be a good beginning for keeping fit.

Moreover, you may devote about 30 minutes to physical activities 3–4 times a week just to improve your style of life. These may be some aerobic exercises, walking, cycling, jogging, swimming or dancing. Healthy way of life is a key to success.

***

- Hello, Miss Stewart! How are you doing? Is anything wrong with you?

- Not so good, doctor. I'm having some problems with my teeth and a bad toothache as well.

- Well, nothing unusual for such a sweet tooth like you. OK. Let me check.

- Is it so bad, doctor?

- You've got another set of cavities and a chipped tooth. Besides, you have cut your third molars or so called wisdom teeth.

- Yes, I got my tooth chipped last night when eating almonds. As for the cavities… How come? I've been taking proper care of my teeth.

- That's because the gaps between your teeth make you prone to frequent cavities. Remember that I always ask you to visit the dentist once every 6 months to maintain your oral hygiene. And you should brush your teeth at least twice a day.

- Are the cavities really awful?

- Well. I'll fix them up with composite filling. And I am sure you are pretty tired of this tartar on your teeth too. I suggest you a special treatment to remove it. I can fix that with scaling and polishing.

- I don't mind. How much will it cost?

- The whole treatment takes 3 sittings, 45 minutes each. Each sitting will cost you 50 dollars.

- Is the procedure painful?

- Not at all. It's absolutely painless.

- That's great! Can I make an appointment for the first sitting then?

- Sure. You can do it at the reception later. Well. Let me put a crown over your chipped tooth.

- Will it be painful?

- It could be a bit painful. We can use anaesthesia to numb your mouth if you want.

- That's OK. I hope I can bear a little pain.

- OK then. Let's get started.

***

- Next, please. Come in… Take a seat.
- Hello, doctor. Last time I came to see you a year ago. You gave me a complete medical check-up then.
- OK. And what is the matter with you at the moment? Any complaints?
- I don't feel very well. I've had an awful headache for 2 days already. Besides I've got a sore throat.
- Have you got a high temperature?
- I took my temperature this morning. It was 37.9.
- That's not so much, I must say.
- So much the better, doctor. If it were over 38 degrees I'd be in bed now.
- Are you coughing much?
- A little bit. I don't have any fits of coughing but I feel pain when I talk and swallow.
- I see. I have to examine your throat and sound your lungs now… Please, strip to the waist. Now take a deep breath. OK. You can breathe out. And now I want you to cough. Good… Well, don't worry, it's just a throat infection. There's nothing wrong with your lungs. I'll prescribe some pills which you should take twice a day, in the morning and in the evening, after your meals. You should also take cough syrup, 3 teaspoonfuls a day will be enough. And it goes without saying that you must not drink any cold liquids.
- All right. Is that all doctor?
- Oh, yes, I shall not prescribe any antibiotics for you, as you haven't got bronchitis fortunately. If you follow my directions, you'll avoid any complications and feel much better in a couple of days. However if the symptoms do not disappear by Thursday you should come and consult me again. So, get well. Here is your prescription. By the way, shall I write out a sick-list?
- No, it's all right. I'm on vacation now. Thank you, doctor. Bye-bye.

***

- Doctor: What seems to be the trouble?
- Patient: I`ve got a cough and a headache. I`m all feverish.
- Doctor: Did you take your temperature?
- Patient: Yes, it is 38.8°C.
- Doctor: How long have you been feeling this way?
- Patient: This way I have been feeling since yesterday. It is flu, isn`t it?
- Doctor: I`m afraid that`s what it is. And no wonder with so much flu about. It is very contagious disease. Why didn`t you call me yesterday?
- Patient: I thought I would be all right.
- Doctor: I will write out a prescription. This medicine is for your headache and it will send your temperature down. Take it every four hours after meals. You can buy the medicine at the nearest chemist`s round the corner.
- Patient: Will the flu last long, doctor?
- Doctor: I hope you will be well again in a week`s time, but I expect you to follow all my instructions. As soon as you feel better come to my surgery for a follow-up examination. If you don`t feel better in a week call me again.
- Patient: Thank you, doctor.
- Doctor Not at all.

***

- Patient: Good morning, doctor.
- Doctor: Good morning. How do you feel today?
- Patient: I feel better today, but I am still of-colour. I`m sleeping badly and I have no appetite.
- Doctor: A little run-down, I think. Have you checked your temperature today?
- Patient: Yes, my temperature is quite normal.

- Doctor: Get to that couch please. I will sound your chest and take your blood pressure. It`s quite normal. Luckily for you there is nothing seriously wrong with your health. You should have a good rest. Can you go on holiday now?
- Patient: I`m going to the sea in two weeks.
- Doctor: That is just what I wanted to recommend.
- Patient: Thank you very much, doctor. I will do it.
- Doctor: Good-bye.
- Patient: Good-bye.

*Игра 2. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить сообщение о сложносочиненном предложении с определенными союзами. Затем каждая группа представляет свое сообщение, сопровождая его максимальным количеством примеров.*

**Карточка 1**

Also - I live in this house, he **also** lives here.
And - You **and** I are busy. You are busy **and** I am free.
As - As you know I live in Saint-Petersburg. As he was busy he couldn't come. As I spoke to him, he was silent. He works **as** an engineer.
either… or - In the evening **I'll either** work at home **or** at the library.
However - He lost his book, but he found it **however** next day. However busy I may be, I'll help you.
If - If I see him I'll speak to him.

**Карточка 2**

As…. as - I am **as** busy **as** you are.
As far as - As **far as** I know he is an engineer.
As long as - As **long as I** stay here we'll often meet.
Neither … nor - We found him **neither** at home, **nor** at the library.
Nevertheless - The work was difficult **nevertheless** we did it.
No matter where - **No matter where** he goes, he takes coffee with him.
No matter how - No **matter how** he tried, he could do nothing.

**Карточка 3**

As well as - This book is useful **as well as** interesting.
Because - I have come here **because I** wanted to help you.
Because of - I was late for the concert **because of** you.
Both …. and - **Both** students **and** teachers were present at the meeting.
But - He invited me **but** I was busy. I know all these words **but** one.
Otherwise - We must start now **otherwise** we might be late.
So … that - There were **so** many people in the hall, **that** I didn't find the seat.
So … as - I came earlier **so as** to speak to you-

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 7. РОЛЕВАЯ ИГРА «В ОФИСЕ», «АНАЛИЗ ПРИРОДНЫХ ЗОН АНГЛОГОВОРЯЩИХ СТРАН»

***Цель работы***: *усвоение лексики по теме: «Деловое общение», «Окружающая среда», «Природные зоны»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить диалог двух коллег в офисе. При подготовке студенты используют приведенные ниже диалоги и тексты*

- My name is Martha Glass. I'm thirty-nine years old and I'm a doctor. I chose the medical profession because I wanted to help people and at the same time make good money. When I was younger I wanted to become a teacher or a nurse, but I soon realized there wasn't much money in either of those professions.
- My parents almost didn't help me, because they didn't want me to have a career at all. They wanted me to do what so many other girls did. They wanted me to become a secretary, marry the boss, have kids and stay at home. Well, I got married, and I had kids, but I have my career as well.
- Hi. I'm Billy. 1 left school when I was sixteen. I didn't have any qualification. I just wanted to earn some money. I got a job in a factory. I didn't mind being a blue-collar worker. All I wanted was enough money to take my girlfriend out on a Saturday night. But then they got robots in to do my job and I was out of work.
- I was out of work for sixteen months. It's terrible being unemployed. The days seem so long. I finally got a job as an unskilled labourer, working for a builder. I'm twenty-five now. I suppose I should go to night classes and get some extra training so that I can earn more money as a skilled worker can.
- My name is George Rushton. I'm a businessman. I'm fifty years old and I've been working for the same company for twenty-five years. I think I've had a very successful career. I started work with the company as a poorly paid clerk. I was one of those nine-to-five white-collar office workers who spend all day with a pencil in one hand and a telephone in the other.
- I hated it. So I was transferred to sales and became one of the company's sales representatives. I travelled all over the country selling the company's products and became the most successful salesperson on the staff. In ten years I have been promoted to manager of the sales department. In another ten years I hope to retire with a good pension.

**In the Office and in the Factory**

**Things in the office**

File, calendar, notice board, computer, monitor, keyboard, filing cabinet, desk, diary, calculator, drawers, wastepaper basket, briefcase.

**Office work**

Brenda works for a company, which produces furniture. She works in an office, which is just opposite the factory where the furniture is made. This is how she spends her day:

She works at a computer most of the time, where she writes letters and reports.

She answers phone calls, mostly from retailers. (= shops selling the factory's furniture)

She makes phone calls to retailers, and the factory making the furniture.

She sends invoices to customers. (= paper showing products sold and the money to pay)

She shows visitors around the factory.

She does general paperwork, e.g. filing reports, writing memos, answering letters.

She arranges meetings for her boss and other managers in the company.

**The 'shop floor' of the factory**

This is where products are manufactured (= made). Modern factories have fewer workers than in the past – this is because of automation (= machines do most of the work), and most factories use an assembly line (= an arrangement in which each worker makes a part of the product and then passes it on to the next person or machine). On an assembly line, workers fit/assemble the different parts, and supervisors (= people in charge/control) check/inspect/examine each stage to make sure the product meets the required standard (= is good enough).

**Finished goods**

Goods (pi) is the general word used for things that are made to be sold. When the product, e.g. a radio, is finished, it is packaged (= put in plastic and then in a box) and stored (= kept) in a warehouse. When a customer, e.g. an electrical shop, orders some of these goods, they are delivered to the shop (= taken to the shop) using road or rail.

**Job Interview**

- Good morning, Miss Jones. So you applied for a job in our team. Am I right?
- Yes, I did. I sent my resume for a position of a restaurant manager.
- That`s good. I`d like to know a bit more about you. Probably you could tell us about your education first.
- Well, I left school at 17 and then for the next five years I studied at Kazan Federal University. I graduated the Department of economics with high honors and was qualified as a manager of enterprise. And after that I did a one-year computer course.
- Well. Your education sounds great, Miss Jones. And have you got any experience? Have you worked before?
- Certainly. First I worked as a manager at children's clothes shop. I stayed there for four years and then I moved on to my present company. They offered me a job of a manager in a big cafe.
- That`s very interesting. Why aren't you happy with your present job, Miss Jones? Why are you going to leave them?
- Well. The salary isn't so bad, I must admit. But the work schedule isn't convenient for me. And I often do a lot of overtime there. Besides you have an excellent reputation and I hope to have more opportunity and growth potential in your company.
- I see. Do you mind business trips? And are you fluent in Italian or German?
- Oh, foreign languages are my favorites. We did Italian and German at the University and I use them when I travel.
- Very good. Can you tell me about your good points then?
- Well… I start my work on time. I learn rather quickly. I am friendly and I am able to work under pressure in a busy company.
- OK. That's enough I think. Well, Miss Jones. Thank you very much. I am pleased to talk to you and we shall inform you about the result of our interview in a few days. Good-bye.

*Игра 2. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить сообщение об одной из природных зон какой-либо англоговорящей страны. Далее каждая группа, используя Интернет, тексты, приведенные ниже, и материалы учебника готовит сообщение. Затем каждая группа представляет свою зону и отвечает на вопросы студентов и преподавателя.*

- Hey, guys, we' re lucky today. The weather is fine. It's ideal for our picnic.

- Yes, Mark. The sky is clear today and the breeze is so gentle. What could be better! Let's sit on the blanket and eat our sandwiches and fruit.

- Is it always like that in this place, Polly?

- Certainly not. It depends on a season and on a month. It's spring, the middle of May now. So it could be damp, rainy now and even stormy sometimes.

- You don't say so! Stormy? Do you mean thunderstorms with lightnings?

- Oh yes. And clouds and heavy showers as well. But most of the time the weather in spring is wonderful and quite warm, with a lot of sunshine. Summers are always sunny and hot here.

- Summer is my favourite season. The weather is absolutely fantastic in my country and the nature is fabulous! And how about autumns?

- In autumn it's windy, chilly, wet and grey as a rule. The temperature can drop to zero at nights. As for winters… well… They are always different. It could be rather mild this year but extremely cold – the next year.

- Do you have much snow? It is my dream to celebrate Christmas when there is white snow outside. It feels like a fairy-tale.

- Then you should come here for your Christmas vacation. We have much snow and sometimes even terrible snowstorms. Kids have much fun making snowmen and playing snowballs. But I prefer to stay indoors in winter, I barely go out.

- Thank you for the invitation, Polly.

***

**Climate and Natural Resources**

The United States of America is a very diverse country. Its nature, climate, population varies from the East Coast to the west, from the northern border to the southern.

Climate is mostly temperate, but tropical in Hawaii and Florida, arctic in Alaska, semiarid in the Great Plains west of the Mississippi River, and arid in the southwest.

Natural resources include coal, copper, lead, molybdenum, phosphates, uranium, bauxite, gold, iron, mercury, nickel, silver, tungsten, zinc, petroleum, natural gas, and timber.

Natural hazards are a great deal of problems for the USA. Every year, they loose hundred millions of dollars, because of natural hazards. The USA is famous for hurricanes along the Atlantic and the Gulf of Mexico coasts and tornadoes in the Midwest and southeast; mud slides in California; forest fires in the west; flooding.

Sometimes there are tsunamis, volcanoes and earthquakes happen. Earthquakes are very often in California.

Talking about environment, one should add that air pollution results in acid rains in both the US and Canada. The US is the largest single emitter of carbon dioxide from the burning of fossil fuels.

Water pollution from runoff of pesticides and fertilizers takes place here.

**Climate and Nature of the USA**

The USA is situated in the central part of the North American Continent. It is washed by the Atlantic Ocean in the east, by the Pacific Ocean in the west and by the Gulf of Mexico in the south. The climate varies from moderate to subtropical. Along the Pacific and Atlantic coasts it is oceanic.

Most of the USA territory is marked by sharp differences between winter and summer. Average winter temperature is about 25 degrees below zero in Alaska and up to 20 degrees above zero in Florida. Average summer temperature varies from 14 degrees above zero in the western part and up to 32 degrees above zero in the southeast. The largest amount of rainfall is noted in Alaska and the southwest of the country. In winter the northern part of the USA usually has a steady snow cover.

The largest rivers of the USA are the Mississippi, the Missouri, the Yukon, the Columbia, and the Colorado. The Great Lakes are situated in the northeast of the country. The region of the Cordilleras has semideserts, while the rest of the territory is rich in forests.

In California, where the climate is usually mild, the famous fruit-raising area is located. Californian oranges, grapefruit and lemons are sold all over the USA and other parts of the world.

The plains of Wyoming, stretching for hundreds of miles, are covered with short grass and sagebrush. This is the land of cattle- and sheep breeding. The south of the country has been an agricultural region for many years. It raises the nation's cotton and tobacco. The USA also grows wheat, corn and different vegetables.

There are a lot of national parks in the USA, the aim of which is to preserve the beauty and treasures of the nature.

**Traditional Regions of the United States**

The differences among Americas traditional regions, or culture areas, tend to be slight and shallow as compared with such areas in most older, more stable countries. The nature of interregional differences can be ascribed to the relative newness of American settlement, a perpetually high degree of mobility, a superb communications system, and the galloping centralisation of economy and government.

Yet, in spite of the nationwide standardisation in many areas of American thought and behaviour, the lingering effects of the older culture areas do remain potent. In the case of the South, for example, the differences helped to precipitate the gravest political crisis and bloodiest military conflict in the nations history.

More than a century after the Civil War, the South remains a powerful entity in political, economic and social terms, and its peculiar status is recognised in religious, educational, athletic and literary circles.

Even more intriguing is the appearance of a series of essentially 20th century regions. Southern California is the largest region, and its special culture has attracted large numbers of immigrants to the state. Similar trends are visible in southern Florida; in Texas, and to a certain degree in regions of New Mexico and Arizona as well.

At the metropolitan level, it is difficult to believe that such distinctive cities as San Francisco, Las Vegas, Dallas, Tucson and Seattle have become like all other American cities. A detailed examination, however, would show significant if sometimes subtle interregional differences in terms of language, religion, diet, folklore, folk architecture and handicrafts, political behaviour, social etiquette and a number of other cultural categories.

**Climate in Great Britain**

The British Isles which are surrounded by the ocean have an insular climate.

There are 3 things that chiefly determine the climate of the United Kingdom: the position of the islands in the temperate belt; the fact that the prevailing winds blow from the west and south-west and the warm current – the Gulf Stream that flows from the Gulf of Mexico along the western shores of England. All these features make the climate more moderate, without striking difference between seasons. It is not very cold in winter and never very hot in summer.

So, the British ports are ice-free and its rivers are not frozen throughout the year. The weather on the British Isles has a bad reputation. It is very changeable and fickle. The British say that there is a climate in other countries, but we have just weather. If you don't like the weather in England, just wait a few minutes.

It rains very often in all seasons in Great Britain. Autumn and winter are the wettest. The sky is usually grey and cold winds blow. On the average, Britain has more than 200 rainy days a year. The English say that they have 3 variants of weather: when it rains in the morning, when it rains in the afternoon, and when it rains all day long. Sometimes it rains so heavily, that they say «It's raining cats and dogs».

Britain is known all over the world for its fogs. Sometimes fogs are so thick that it's impossible to see anything within a few meters. The winter fogs of London are, indeed, awful; they surpass all imagination. In a dense fog all traffic is stopped, no vehicle can move from fear of dreadful accidents. So, we may say that the British climate has three main features: it is mild, humid and very changeable.

The weather in England is very changeable. A fine morning can change into a wet afternoon and evening. And a nasty morning can change into a fine afternoon. That is why it is natural for the English to use the comparison "as changeable as the weather" of a person who often changes his mood or opinion about something. "Other countries have a climate; in England we have weather". This statement is often made by the English to describe the meteorological conditions of their country.

The English also say that they have three variants of weather: when it rains in the morning, when it rains in the afternoon, or when it rains all day long.

The weather is the favorite conversational topic in England. When two Englishmen meet, their first words will be "How are you?" And after the reply "Very well, thank you; how are you?" the next remark is almost certain to be about the weather. When they go abroad the English often surprise people of other nationalities by this tendency to talk about the weather, a topic of conversation that other people do not find so interesting.

The best time of the year in England is spring (of course, it rains in spring, too). The two worst months in Britain are January and February. They are cold, damp, and unpleasant. The best place in the world then is at home by the fire.

Summer months are rather cold and there can be a lot of rainy days. So most people, who look forward to summer holidays, plan to go abroad for the summer, to France or somewhere on the Continent.

The most unpleasant aspects of the weather in England are fog and smog.

While some countries have too much history, Canada has too much geography.

**From Sea to Sea**

Occupying the northern half of the North American continent, Canada has a landmass of nearly 10 million km2.

Canada's motto, 'From Sea to Sea', is geographically inaccurate. In addition to its long coastlines on the Atlantic and Pacific, Canada has a third sea coast on the Arctic Ocean, giving it the longest coastline of any country.

To the south, Canada shares an 8,892-km boundary with the United States. To the north, the Arctic islands come within 800 km of the North Pole. Canada's neighbour across the frozen Arctic Ocean is Russia.

**A Long Thin Band**

Because of the harsh northern climate, Only 12 per cent of the land is suitable for agriculture. Thus, most of the population of 26 million live in cities within a few hundred kilometres of the southern border – where the climate is milder – in a long thin band stretching between the Atlantic and the Pacific oceans.

**Numberless Lakes and Great Rivers**

It has been estimated that Canada has one-seventh of the world's fresh water. In addition to sharing the Great Lakes with the United States, Canada has many other freshwater seas and mighty rivers.

**The Pacific Coast**

Bathed by warm, moist Pacific air currents, the British Columbia coast, indented by deep fiords and shielded from the Pacific by Vancouver Island, has the most moderate.

**The Cordillera**

Canada's highest peaks, however, are not in the Rockies, but in the St. Elias Mountains, an extension of the Cordillera stretching north into the Yukon and Alaska. The highest point in Canada, Mt. Logan (6,050 m).

**The Prairies**

The plains of Alberta, Saskatchewan and Manitoba are among the richest grain-producing regions in the world.

Yet even here are surprises. If you drive north, you descend into the Red Deer River valley. Here, in desert-like conditions, water and wind have created strange shapes in the sandstone called 'hoodoos'. The same forces of erosion have uncovered some of the largest concentrations of dinosaur fossils, examples of which are displayed in museums in Canada and around the world.

Alberta is Canada's leading producer of petroleum. The sedimentary rocks underlying the Prairies have important deposits of oil, gas and potash.

**The Canadian Shield**

Look at a map of Canada and you will see a huge inland sea called Hudson Bay. Wrapped around this bay like a horseshoe is a rocky region called the Canadian Shield.

The region is a storehouse of minerals, including gold, silver, zinc, copper and uranium, and Canada's great mining towns are located here – Sudbury and Timmins in Ontario, Val d'Or in Quebec, and Flin Flon and Thompson in Manitoba.

**Great Lakes – St. Lawrence Lowlands**

Southern Quebec and Ontario, the industrial heartland of Canada, contain Canada's two largest cities, Montreal and Toronto. In this small region, 50 per cent of Canadians live and 70 per cent of Canada's manufactured goods are produced.

The region also has prime agricultural land. The Niagara Peninsula, for example, has some of the best farmland in Canada. The large expanses of lakes Erie and Ontario extend the number of frost-free days, permitting the cultivation of grapes, peaches, pears and other soft fruits.

The region is sugar maple tree country. In the autumn, the tree's leaves – Canada's national symbol – are ablaze in red, orange and gold. The sap is collected in spring and evaporated to make maple syrup and sugar, a culinary delicacy first used by the aboriginal North American peoples.

**Atlantic Provinces – Appalachian Region**

New Brunswick, Nova Scotia, Prince Edward Island and Newfoundland are the smallest Canadian provinces, and the first to be settled by Europeans.

The shallow continental shelf extends 400 km off the east coast of Newfoundland where the mixing of ocean currents has created one of the richest fishing grounds in the world.

Agriculture flourishes in the fertile valleys, such as the Saint John River Valley, New Brunswick, and the Annapolis Valley, Nova Scotia.

Prince Edward Island in the Gulf of St. Lawrence is famous for its potatoes. This fertile island is Canada's smallest province, making up a mere 0.1 percent of Canada's landmass.

**The Arctic**

North of the tree-line is a land of harsh beauty. During the short summer, when daylight is nearly continuous and a profusion of flowers blooms on the tundra, the temperature can reach 30 °C. Yet the winters are long, bitterly cold, dark and unforgiving.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 8. КОНКУРС ПРОЕКТОВ «ВИДЫ ПРЕДПРИЯТИЙ», «ГОРОДА США»

***Цель работы***: *усвоение лексики по теме: «Город», «Природные зоны»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Проект. Все студенты заранее получили задание подготовить сообщение об одном из предприятий или городов США, включающем письменный текст, устный доклад и инфографику. На занятии каждый представляет свой проект и отвечает на вопросы студентов и преподавателя.*

**Advertisement as a service**

Although the average citizen is usually annoyed by all the advertisements printed in newspapers and magazines and the commercials broadcast on TV, the impact of the whole advertising industry on a single person is immense and plays a very important role in our lives. Advertising absorbs vast sums of money but it is useful to the community. What are the functions of advertisements? The first one to mention is to inform. A lot of the information people have about household devices, cars, building materials, electronic equipment, cosmetics, detergents and food is largely derived from the advertisements they read. Advertisements introduce them to new products or remind them of the existing ones. The second function is to sell. The products are shown from the best point of view and the potential buyer, on having entered the store, unconsciously chooses the advertised products. One buys this washing powder or this chewing gum, because the colorful TV commercials convince him of the best qualities of the product. Even cigarettes or sweets or alcohol are associated with the good values of human life such as joy, freedom, love and happiness, and just those associations make a person choose the advertised products. The aim of a good advertisement is to create a consumer demand to buy the advertised product or service. Children are good example as they usually want the particular kind of chocolate or toy or chewing-gum. Being naive they cannot evaluate objectively what is truthful and what is exaggerated and select the really good products unless they buy the goods and check for themselves. Thirdly, since the majority of advertisements are printed in our press we pay less for newspapers and magazines, also TV in most countries in cheap. The public advertising seen on street hoardings, railway stations and buildings makes people's life more joyful. Moreover, all those small ads in the press concerning "employment", "education" and "For sale and wanted" columns, help ordinary people to find a better job or a better employee, to sell or to buy their second-hand things and find services, or learn about educational facilities, social events such as, concerts, theatre plays, football matches, and to announce births, marriages and deaths. Thus despite our dissatisfaction when being bombarded by all the advertisers' information we must admit that they do perform a useful service to society, and advertisements are an essential part of our everyday life.

**Brands**

What is a brand? In my opinion, it's not only a trademark of some company, but the name of certain product we use every day. For example, speaking about coffee most of us say Nescafe, but not 'coffee'. This short example also illustrates the main aim of producers - to create brand popularity, so that most of people would recognize the product among the competitors products. Advertising campaigns are launched to enhance brand awareness, that's why sometimes brand costs more than the whole company, for example one day of advertising at Yandex website (what is called by Yandex sales managers as 'increasing brand popularity') costs $20000. Recognition of a brand or, how it's

called, brand awareness helps people to find the necessary size, quantity, taste, especially, when they are in another country and don't know the local products' specifications. What qualities should brand name possess? First of all, it should be eye-catching. NameLab, company, which creates brand names, gives an example of 7-Up Company, which lost $120 millions using name 'Lyke Cola' as a brand name first time after launching its product. Lexicon Company was more original, creating brand name 'Pentium' for the Intel Processor: "We've got '-ium' from the scientific text - founder of Lexicon says, - and multiplied it with 'pent'. It sounded very strong, like a real chemical element." Name Sony is based on 'son', which means sound in most of the countries. As all brand names are registered and protected by law, no one else can produce the same product under such brand name. It's a very hard to create a new brand name, as more than 365000 brands were registered in October, 2000 by American Patent Organization, whereas Oxford dictionary consists of 615100 words, so some companies use brand stretching - using a leader-brand to launch a new product in a new category, e.g. 'Bochkarev' chips. Brands always add value to products. That's why branded products seem to be more expensive among other ones. But if we pay more, we pay for better quality. All in all, brands are one of the moving forces of globalisation.

**Employment**

Getting a job is a very hard period in the life of most people. Companies choose an employee from hundreds of candidates according to special rules, that's why there're special 'typical' factors, influencing on employer's choice. Among such factors are: age, sex, experience, family background and marital status, personality and references. If you're to go to an interview tomorrow, sleep well before it and don't forget your CV at home - is the basic rule. Moreover, there're some recommendations, which can help you, for example, to read annual report, or company newspaper of the company to show your understanding of the corporate strategy on the interview. What's more, you should choose corresponding dress code for the interview. Even such advices are to help you make a good impression; some companies don't want to hire a man, who follows every advice. To illustrate this, I can quote Artemiy Lebedev, the most famous Russian web-designer: "If you enclose a standard stupid resume, written by the rules of American bureaucracy, we would delete it immediately after receiving. If your CV is composed according to all rules, we wouldn't choose you, as we might think, that your profession is to acquire a job". After getting a job, you may have some unexpected troubles with boss, too: e.g. if you dye your hair or wear something not appropriate. The best solution of such situation is to ask a trade union for advice, which can always help you in your fight with an employer. Of course, if you affect company discipline not coming in time or working badly, your dismissal wouldn't be unfair. To conclude, I can say that it is sometimes hard not only to get a job, but also to work in the staff, and if you don't want to be laid off, you should follow company rules, it is a must.

**The Base of Industry**

Americas heavy industry depends upon three resources: iron ore from the Lake Superior area, coal from western Pennsylvania, and transportation across the Great Lakes District.

Steel making is basic, but there are many other related industries in this area, too; glass, nonferrous metals, chemicals, rubber, and machine-building industries.

Pittsburgh is the first of the great steel cities.

The other great steel-making centres are Chicago, Detroit, Youngtown, Cleveland, Toledo, Erie, Buffalo.

Detroit is the heart of the automobile industry. It began as a wagon-making town.

The cargo tonnage which passes between Lake Superior and Lake Huron almost equals the combined capacity of the Panama and Suez Canals.

**Industry**

Iron ore is mined in Northamptonshire and Humberside.

Cornwall is the only county in England that provides the nation with tin ore.

Sand, gravel, widely available, provide raw materials for the construction industry.

Clay and salt are found in the northwestern England, and china clay is available in Cornwall.

More than two-thirds of those employed in England work in the service industries.

London is a major financial, banking, and insurance centre.

Cambridge, Ipswich, and Norwich are important service and high-tech centres.

Nearly a quarter of England's workers are employed in manufacturing. Major industries located in the northern counties include food processing, brewing, and the manufacture of chemicals, textiles, computers, automobiles, aircraft, clothing, glass, and paper products.

Leading industries in southeastern England are' pharmaceuticals, computers, microelectronics, aircraft parts, and automobiles.

England produces 90 % of Britain's coal.

**San Francisco**

European discovery and exploration of the San Francisco Bay area and its' islands began in 1542. In 1579, Sir Francis Drake and his crew arrived in Golden Hind and spent five weeks repairing the ship and meeting with the natives. The Spanish found the entrance to the bay in 1769, and by 1776, the first colonizing party arrived to found the San Francisco and Mission Dolores.

In 1869, the first train arrived in San Francisco and in 1870 San Francisco became the tenth largest city in the United States. A large Chinese population of labourers recruited in the 1840s and 1850's settled there. Irish immigrants settled into the Mission area and French, Italian, German, Russian, Australian, Jewish and many other nationalities contributed to the city's development and growing.

San Francisco was a tiny settlement before the Gold Rush of 1849. The Gold Rush brought wild crowds of people to the city and surroundings.

After the rush was over, many prospectors returned from the gold fields and settled in the city, realizing that fortunes could be made just as well there. Mercantile establishments, small industries, and shipping to the Orient brought prosperity to the newcomers. San Francisco attracted a colourful array of characters. Famous writers such as Jack London and Mark Twain were there.

The 1906 Earthquake and fire devastated the city. But with its spirit, the city rebuilt itself–into a grander city than even before. And it was no surprise that there is the Golden Gate Bridge–one of the world's longest suspension bridges – over icy-cold, shark infested bay. It has the highest bridge towers ever made.

San Francisco hosts over 16 million people every year. Everyone knows about the Golden Gate Bridge, Alcatraz and Chinatown.

San Francisco is a popular location any time of the year. Summer is the prime tourist season, so prices are higher, lines are longer. One can go to most of the popular destinations: Union Square, North Beach, Chinatown, Ghirardelli Square and the Financial District.

**Philadelphia**

Philadelphia is situated in the east of the USA. It is one of the few large cities in the United States to have an old and well-preserved city centre.

Philadelphia is an important city for American history: it was in fact to be the first capital city of the colonial states from 1790 till 1800 after their rebellion against the British government as well as being the birthplace of several famous men like Franklin, Jefferson and Washington.

In any case, «old», in the United States means that the historic buildings mostly date from the 18th century at the earliest. By 1774 Philadelphia had become the military, economic, and political centre of the colonies. The USA constitution was the first written constitution in the world adopted in this city in 1787. The Declaration of Independence was also proclaimed in 1776 here.

Many U.S. «firsts» were associated with the city of Philadelphia: first public school was opened in 1689. State's first newspaper was published here in 1719, America's first hospital was opened in 1755, and first American flag firstly appeared in Philadelphia in 1777.

There are many places of interest in Philadelphia, for example, the Independence National Historical Park. We can see many monuments there. One of them is the Liberty Bell. Now the Liberty Bell is a symbol of freedom. The sound of this Bell told people about the first public reading of the Declaration of Independence. It was in July, 1776.

Philadelphia is one of the cultural centres of the country. The Parkway is the cultural centre of Philadelphia. One can see the Philadelphia Museum of Art which is one of the greatest art museums in the world, College of Art, Academy of Sciences and the Academy of Fine Arts in Philadelphia.

There are many hotels, theatres, shops and museums in Philadelphia.

There is the Pennsylvania University in Philadelphia. This University has an interesting and big library.

Philadelphia is a beautiful city with many skyscrapers.

Today, Philadelphia's economy is one of the most diverse in the United States. It is based on a system of manufacturing, commercial, and technological activities, and on tourism. In the downtown area, there are many headquarters for major regional, national, and international corporations.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9. КОНКУРС ПРОЕКТОВ «РАБОТА АКЦИОНЕРНОГО ОБЩЕСТВА»

***Цель работы***: *усвоение лексики по теме: «Предприятие», «Формы предприятий»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Проект. Все студенты заранее получили задание подготовить сообщение об одном из акционерных обществ, включающем письменный текст, устный доклад и инфографику. На занятии каждый представляет свой проект и отвечает на вопросы студентов и преподавателя.*

**Company History:**

Apple Computer, Inc. is largely responsible for the enormous growth of the personal computer industry in the 20th century. The introduction of the Macintosh line of personal computers in 1984 established the company as an innovator in industrial design whose products became renowned for their intuitive ease of use. Though battered by bad decision-making during the 1990s, Apple continues to exude the same enviable characteristics in the 21st century that catapulted the company toward fame during the 1980s. The company designs, manufactures, and markets personal computers, software, and peripherals, concentrating on lower-cost, uniquely designed computers such as iMAC and Power Macintosh models.

**Origins**

Apple was founded in April 1976 by Steve Wozniak, then 26 years old, and Steve Jobs, 21, both college dropouts. Their partnership began several years earlier when Wozniak, a talented, self-taught electronics engineer, began building boxes that allowed him to make long-distance phone calls for free. The pair sold several hundred such boxes.

In 1976 Wozniak was working on another box--the Apple I computer, without keyboard or power supply--for a computer hobbyist club. Jobs and Wozniak sold their most valuable possessions, a van and two calculators, raising $1,300 with which to start a company. A local retailer ordered 50 of the computers, which were built in Jobs's garage. They eventually sold 200 to computer hobbyists in the San Francisco Bay area for $666 each. Later that summer, Wozniak began work on the Apple II, designed to appeal to a greater market than computer hobbyists. Jobs hired local computer enthusiasts, many of them still in high school, to assemble circuit boards and design software. Early microcomputers had usually been housed in metal boxes. With the general consumer in mind, Jobs planned to house the Apple II in a more attractive modular beige plastic container.

Jobs wanted to create a large company and consulted with Mike Markkula, a retired electronics engineer who had managed marketing for Intel Corporation and Fairchild Semiconductor. Chairman Markkula bought one-third of the company for $250,000, helped Jobs with the business plan, and in 1977 hired Mike Scott as president. Wozniak worked for Apple full time in his engineering capacity.

Jobs recruited Regis McKenna, owner of one of the most successful advertising and public relations firms in Silicon Valley, to devise an advertising strategy for the company. McKenna designed the Apple logo and began

advertising personal computers in consumer magazines. Apple's professional marketing team placed the Apple II in retail stores, and by June 1977, annual sales reached $1 million. It was the first microcomputer to use color graphics, with a television set as the screen. In addition, the Apple II expansion slot made it more versatile than competing computers.

The earliest Apple IIs read and stored information on cassette tapes, which were unreliable and slow. By 1978 Wozniak had invented the Apple Disk II, at the time the fastest and cheapest disk drive offered by any computer manufacturer. The Disk II made possible the development of software for the Apple II. The introduction of Apple II, with a user manual, at a consumer electronics show signaled that Apple was expanding beyond the hobbyist market to make its computers consumer items. By the end of 1978, Apple was one of the fastest-growing companies in the United States, with its products carried by over 100 dealers.

In 1979 Apple introduced the Apple II+ with far more memory than the Apple II and an easier startup system, and the Silentype, the company's first printer. VisiCalc, the first spreadsheet for microcomputers, was also released that year. Its popularity helped to sell many Apple IIs. By the end of the year sales were up 400 percent from 1978, at over 35,000 computers. Apple Fortran, introduced in March 1980, led to the further development of software, particularly technical and educational applications.

In December 1980, Apple went public. Its offering of 4.6 million shares at $22 each sold out within minutes. A second offering of 2.6 million shares quickly sold out in May 1981.

Meanwhile Apple was working on the Apple II's successor, which was intended to feature expanded memory and graphics capabilities and run the software already designed for the Apple II. The company, fearful that the Apple II would soon be outdated, put time pressures on the designers of the Apple III, despite the fact that sales of the Apple II more than doubled to 78,000 in 1980. The Apple III was well received when it was released in September 1980 at $3,495, and many predicted it would achieve its goal of breaking into the office market dominated by IBM. However, the Apple III was released without adequate testing, and many units proved to be defective. Production was halted and the problems were fixed, but the Apple III never sold as well as the Apple II. It was discontinued in April 1984.

The problems with the Apple III prompted Mike Scott to lay off employees in February 1981, a move with which Jobs disagreed. As a result, Mike Markkula became president and Jobs chairman. Scott was named vice-chairman shortly before leaving the firm.

Despite the problems with Apple III, the company forged ahead, tripling its 1981 research and development budget to $21 million, releasing 40 new software programs, opening European offices, and putting out its first hard disk. By January 1982, 650,000 Apple computers had been sold worldwide. In December 1982, Apple became the first personal computer company to reach $1 billion in annual sales.

The next year, Apple lost its position as chief supplier of personal computers in Europe to IBM, and tried to challenge IBM in the business market with the Lisa computer. Lisa introduced the mouse, a hand-controlled pointer, and displayed pictures on the computer screen that substituted for keyboard commands. These innovations come out of Jobs's determination to design an unintimidating computer that anyone could use.

Unfortunately, the Lisa did not sell as well as Apple had hoped. Apple was having difficulty designing the elaborate software to link together a number of Lisas and was finding it hard to break IBM's hold on the business market. Apple's earnings went down and its stock plummeted to $35, half of its sale price in 1982. Mike Markkula had viewed his presidency as a temporary position, and in April 1983, Jobs brought in John Sculley, formerly president of Pepsi-Cola, as the new president of Apple. Jobs felt the company needed Sculley's marketing expertise.

**1984 Debut of the Macintosh**

The production division for Lisa had been vying with Jobs's Macintosh division. The Macintosh personal computer offered Lisa's innovations at a fraction of the price. Jobs saw the Macintosh as the 'people's computer'--designed for people with little technical knowledge. With the failure of the Lisa, the Macintosh was seen as the future of the company. Launched with a television commercial in January 1984, the Macintosh was unveiled soon after, with a price tag of $2,495 and a new 3-inch disk drive that was faster than the 5-inch drives used in other machines, including the Apple II.

Apple sold 70,000 Macintosh computers in the first 100 days. In September 1984 a new Macintosh was released with more memory and two disk drives. Jobs was convinced that anyone who tried the Macintosh would buy it. A national advertisement offered people the chance to take a Macintosh home for 24 hours, and over 200,000 people did so. At the same time, Apple sold its two millionth Apple II. Over the next six months Apple released numerous products for the Macintosh, including a laser printer and a hard drive.

Despite these successes, Macintosh sales temporarily fell off after a promising start, and the company was troubled by internal problems. Infighting between divisions continued, and poor inventory tracking led to overproduction. Although Jobs had originally been a strong supporter of Sculley, Jobs eventually decided to oust Sculley; Jobs, however, lost the ensuing showdown. Sculley reorganized Apple in June 1985 to end the infighting caused by the product-line divisions, and Jobs, along with several other Apple executives, left the company in September. They founded a new computer company, NeXT Incorporated , which would later emerge as a rival to Apple in the business computer market.

The Macintosh personal computer finally moved Apple into the business office market. Corporations saw its ease of use as a distinct advantage. It was far cheaper than the Lisa and had the necessary software to link office computers. In 1986 and 1987 Apple produced three new Macintosh personal computers with improved memory and power. By 1988, over one million Macintosh computers had been sold, with 70 percent of sales to corporations. Software was created that allowed the Macintosh to be connected to IBM-based systems. Apple grew rapidly; income for 1988 topped $400 million on sales of $4.07 billion, up from income of $217 million on sales of $1.9 billion in 1986. Apple had 5,500 employees in 1986 and over 14,600 by the early 1990s.

In 1988, Apple management had expected a worldwide shortage of memory chips to worsen. They bought millions when prices were high, only to have the shortage end and prices fall soon after. Apple ordered sharp price increases for the Macintosh line just before the Christmas buying season, and consumers bought the less expensive Apple line or other brands. In early 1989, Apple released significantly enhanced versions of the two upper-end Macintosh computers, the SE and the Macintosh II, primarily to compete for the office market. At the same time IBM marketed a new operating system that mimicked the Macintosh's ease of use. In May 1989 Apple announced plans for its new operating system, System 7, which would be available to users the next year and allow Macintoshes to run tasks on more than one program simultaneously.

Apple was reorganized in August 1988 into four operating divisions: Apple USA, Apple Europe, Apple Pacific, and Apple Products. Dissatisfied with the changes, many longtime Apple executives left. In July 1990, Robert Puette, former head of Hewlett-Packard's personal computer business, became head of the Apple USA division. Sculley saw the reorganization as an attempt to create fewer layers of management within Apple, thus encouraging innovation among staff. Analysts credit Sculley with expanding Apple from a consumer and education computer company to a business computer company, one of the biggest and fastest-growing corporations in the United States.

Competition in the industry of information technology involved Apple in a number of lawsuits. In December 1989 for instance, the Xerox Corporation, in a $150 million lawsuit, charged Apple with unlawfully using Xerox technology for the Macintosh software. Apple did not deny borrowing from Xerox technology but explained that the company had spent millions to refine that technology and had used other sources as well. In 1990 the court found in favor of Apple in the Xerox case. Earlier, in March 1988, Apple had brought suits against Microsoft and Hewlett-Packard, charging copyright infringement. Four years later, in the spring of 1992, Apple's case was dealt a severe blow in a surprise ruling: copyright protection cannot be based on 'look and feel' (appearance) alone; rather, 'specific' features of an original program must be detailed by developers for protection.

**Mismanagement--Crippling an Industry Giant: 1990s**

Apple entered the 1990s well aware that the conditions that made the company an industry giant in the previous decade had changed dramatically. Management recognized that for Apple to succeed in the future, corporate strategies would have to be reexamined.

Apple had soared through the 1980s on the backs of its large, expensive computers, which earned the company a committed, yet relatively small following. Sculley and his team saw that competitors were relying increasingly on the user-friendly graphics that had become the Macintosh signature and recognized that Apple needed to introduce smaller,

cheaper models, such as the Classic and LC, which were instant hits. At a time when the industry was seeing slow unit sales, the numbers at Apple were skyrocketing. In 1990, desktop Macs accounted for 11 percent of the PCs sold through American computer dealers. In mid-1992, the figure was 19 percent.

But these modestly priced models had a considerably smaller profit margin than their larger cousins. So even if sales took off, as they did, profits were threatened. In a severe austerity move, Apple laid off nearly ten percent of its workforce, consolidated facilities, moved production plants to areas where it was cheaper to operate, and drastically altered its corporate organizational chart. The bill for such forward-looking surgery was great, however, and in 1991 profits were off 35 percent. But analysts said that such pitfalls were expected, indeed necessary, if the company intended to position itself as a leaner, better-conditioned fighter in the years ahead.

Looking ahead is what analysts say saved Apple from foundering. In 1992, after the core of the suit that Apple had brought against Microsoft and Hewlett-Packard was dismissed, industry observers pointed out that although the loss was a disappointment for Apple, the company wisely had not banked on a victory. They credited Apple's ambitious plans for the future with quickly turning the lawsuit into yesterday's news.

In addition to remaining faithful to its central business of computer making--the notebook PowerBook series, released in 1991, garnered a 21 percent market share in less than six months--Apple intended to ride a digital wave into the next century. The company geared itself to participate in a revolution in the consumer electronics industry, in which products that were limited by a slow, restrictive analog system would be replaced by faster, digital gadgets on the cutting edge of telecommunications technology. Apple also experimented with the interweaving of sound and visuals in the operations of its computers.

For Apple, the most pressing issue of the 1990s was not related to technology, but concerned capable and consistent management. The company endured tortuous failures throughout much of the decade, as one chief executive officer after another faltered miserably. Scully was forced out of his leadership position by Apple's board of directors in 1993. His replacement, Michael Spindler, broke tradition by licensing Apple technology to outside firms, paving the way for ill-fated Apple clones that ultimately eroded Apple's profits. Spindler also oversaw the introduction of the Power Macintosh line in 1994, an episode in Apple's history that typified the perception that the company had the right products but not the right people to deliver the products to the market. Power Macintosh computers were highly sought after, but after overestimating demand for the earlier release of its PowerBook laptops, the company grossly underestimated demand for the Power Macintosh line. By 1995, Apple had $1 billion worth of unfilled orders, and investors took note of the embarrassing miscue. In a two-day period, Apple's stock value plunged 15 percent.

After Spindler's much-publicized mistake of 1995, Apple's directors were ready to hand the leadership reins to someone new. Gil Amelio, credited with spearheading the recovery of National Semiconductor, was named chief executive officer in February 1996, beginning another notorious era of leadership for the beleaguered Cupertino company. Amelio cut Apple's payroll by a third and slashed operating costs, but drew a hail of criticism for his compensation package and his inability to relate to Apple's unique corporate culture. Apple's financial losses, meanwhile, mounted, reaching $816 million in 1996 and a staggering $1 billion in 1997. The company' stock, which had traded at more than $70 per share in 1991, fell to $14 per share. Its market share, 16 percent in the late 1980s, stood at less than four percent. *Fortune* magazine offered its analysis, referring to Apple in its March 3, 1997 issue as 'Silicon Valley's paragon of dysfunctional management.'

Amelio was ousted from the company in July 1997, but before his departure a significant deal was concluded that brought Apple's savior to Cupertino. In December 1996, Apple paid $377 million for NeXT, a small, $50-million-in-sales company founded and led by Steve Jobs. Concurrent with the acquisition, Amelio hired Jobs as his special advisor, marking the return of Apple's visionary 12 years after he had left. In September 1997, two months after Amelio's exit, Apple's board of directors named Jobs interim chief executive officer. Apple's recovery occurred during the ensuing months.

Jobs assumed his responsibilities with the same passion and understanding that had made Apple one of the greatest success stories in business history. He immediately discontinued the licensing agreement that spawned Apple clones. He eliminated 15 of the company's 19 products, withdrawing Apple's involvement in making printers, scanners, portable digital assistants, and other peripherals. From 1997 forward, Apple would focus exclusively on desktop and

portable Macintoshes for professional and consumer customers. Jobs closed plants, laid off thousands of workers, and sold stock to rival Microsoft Corporation, receiving a cash infusion of $150 million in exchange. Apple's organizational hierarchy underwent sweeping reorganization as well, but the most visible indication of Jobs's return was unveiled in August 1998. Distressed by his company's lack of popular computers that retailed for less than $2,000, Jobs tapped Apple's resources and, ten months after the project began, unveiled the massively successful iMAC, a sleek and colorful computer that embodied Apple's skill in design and functionality.

Because of Jobs's restorative efforts, Apple exited the 1990s as a pared-down version of its former self, but, importantly, a profitable company once again. Annual sales, which totaled $11.5 billion in 1995, stood at $5.9 billion in 1998, from which the company recorded a profit of $309 million. In 1999, sales grew a modest 3.2 percent, but the newfound health of the company was evident in a 94 percent gain in net income, as Apple's profits swelled to $601 million. Further, Apples' stock mustered a remarkable rebound, climbing 140 percent to $99 per share in 1999. By the decade's end, 'interim' was dropped from Jobs's corporate title, signaling Jobs's return on a permanent basis and fueling optimism that Apple could look forward to a decade of vibrant and consistent growth.

In the year 2000 Steve Jobs announced that he would become the new CEO of the company and Mitch Mandich who was the former chief sales executive announced that he would be stepping down as well as the announcement of upcoming products and upgrades are provided such as the PowerMac Cube. Apples success continued with the launch of the PowerBook G4 in 2001 which included a series of Notebook home computers. Another great milestone for Apple INC. in 2001 was the launch of the popular iPod which is a small handheld media player. 2001 was the launch year for the OS x operating system. Another important milestone in 2001 was the licensing of Amazonâ€™s 1 Click.

In 2002 Apple teamed up with Sun and Ericsson and the former Vice President of Education John Couch returned as well. Other notable advancements for Apple in 2002 were the acquisition of Magic, a music software company as well as the FireWire Company and the announcement that their retail stores would soon be expanding to include overseas locations. Apple was awarded an Emmy for technology in 2002 and there was also an announcement that Larry Ellison would be resigning form the board.

The CEO, Steve Jobs underwent surgery in 2003 for pancreatic cancer. The new ad campaign which features the musical band U2 was launched in 2004. One of the most exciting advancements of Apple in 2004 was the opening of the iTunes store. A new version of the iPod was also introduced in 2004 and featured the 4th generation iPod as well as the unveiling of the video iPod. In 2005 the release of the IPod Nano was successfully launched and Jeff Raskin who was the computer interface expert for Apple Computers Inc. at the time died from cancer. Further advancements and events in 2005 include the acquisition of Schema Soft as well as the switch to the use of Intel processor chips in Apple products. The success of Apply Computers was apparent with the download of more than one million videos within three weeks of the launch of the Video iPod.

In 2006 Avie Tevanian who was the software development leader for Apple announces his resignation and the announcement of the computer take bake program was also a buzz. The popular MacBook Pro line was also introduced in 2006 and offered a line of portable computers to consumers. Although Apple was already a leader in technology, the release of the iPhone in 2007 brought the company great gains and opened up a whole new world for users due to the sleek interface with a single button that featured a touch screen and virtual keyboard as well as the introduction of Apple TV and the iPod touch which was very similar to the iPhone without the telephone capabilities featuring wireless capabilities.

In 2008 the App Store was unveiled as an iTunes update and featured small applications which could be easily downloaded to your iPhone or iPod. These applications included everything from games to business and social tools. The MacBook air was also released in 2008. 2009 brought some problems for the company when CEO Steve Jobs had to take a leave of absence from the company due to health reasons. After a liver transplant he returned to work that same year.

Later in 2009 the iPhone 3GS was released as the new version of the original iPhone and sales for their iPod reached more than $200 million. In 2012 Cooks who filled in for Jobs during his medical leave was awarded bonus of $22 million dollars for his outstanding leadership during Jobsâ€™ leave of absence in which time Appleâ€™s stock prices increased by almost 70%. In 2010 the new iPad was also launched which features a large 10â€ touchscreen. It

quickly claimed more than 80% of the tablet market by the end of the year. Music from the British band The Beatles became available on iTunes after much debate.

In 2011 the announcement was made that Jobs would take an additional medical leave of absence. The iPhone was now available through Verizon wireless which ended the monopoly which AT&T had with the iPhone due to the expiration of the contract giving AT&T exclusive rights to the sales of the iPhone in the United States. The IPad 2 and iPhone 4 Pro were also introduced in 2011 which offered new innovative features and a more streamlined and sleek design and style.

2011 also brought the launch of the iPhone 4S in October with the introduction of Siri - is a voice control friend which will quickly provide maps, directions, phone calls, and other features by verbal request. Four million units were sold within the first few weeks of release.

2012 brought the release of the release of the new iPhone five in September with more than 5 million being sold within the first 3 days of the release and caused a backorder and delay in shipment because the company did not anticipate the demand.

**Principal Subsidiaries:** Apple Computer, Inc. Limited (Ireland); Apple Computer Limited (Ireland); Apple Computer U.K. Limited (U.K.); Apple Computer International (Ireland); FileMaker Inc.; Apple Japan, LLC; Apple Computer B.V. (Netherlands); A C Real Properties, Inc.

**Principal Competitors:** Compaq Computer Corporation; Dell Computer Corporation; International Business Machines Corporation; Microsoft Corporation; Sun Microsystems, Inc.

## ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 10. ДЕЛОВАЯ ИГРА «ДЕЛОВЫЕ ПИСЬМА»

***Цель работы***: *усвоение структуры и лексики делового письма, видов делового письма; формирование умений построения писем различных видов; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на 2-3 группы. Каждая группа готовит презентацию своей фирмы (название, виды деятельности). Затем группы начинают обмениваться письмами. Виды писем определяются преподавателем.*

E-mail Writing

---

Read the email and decide which parts are

- the intro
- the details
- the action
- the close

Dear Simon,

Thank you very much for showing me round your production facilities. I was most impressed.

I'm pleased to tell you that your company is one of two short-listed for the production of our new website video. This is an important part of our marketing strategy and we are sure you will treat this with the importance it deserves.

I need to have a draft outline of your thoughts for this video by the end of the month. Please send this to me by email as an attachment.

If you need any further help, feel free to contact me.

Best wishes.

Sandy Benny

Marketing Manager

Dear Ms Wager,

YOUR ORDER NUMBER CB4578

Thank you for your email of the 3rd March.

I'm very sorry to hear about the mistake we made with your order. I have investigated this and found an error in our order-processing system.

I have arranged for a repeat order to be sent to you today. I have also enclosed a voucher for you to receive a 5% discount on your next order. For your convenience, I have included a copy of our new catalogue.

Please do not hesitate to email me if you have any further problems.

Once again, please accept my apologies for the inconvenience caused

Yours sincerely,

Carlton Palmer

**Литература**

**Основная учебная**

1. **Резникова, Т.В.** Английский язык. Практика перевода. Часть 2 [Электронный ресурс]: рабочий учебник/Резникова Т.В. - 2012. - http://lib.muh.ru

2. **Бочкарева Т.С.** Английский язык [Электронный ресурс]: учебное пособие по английскому языку/ Бочкарева Т.С., Чапалда К.Г.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2013.— 99 c.— : http://www.iprbookshop.ru/30100.— ЭБС «IPRbooks»

3. **Сохрякова Е.С.** Английский язык [Электронный ресурс]: учебное пособие/ Сохрякова Е.С.— Электрон. текстовые данные.— Омск: Омский государственный институт сервиса, 2014.— 99 c.— : http://www.iprbookshop.ru/26678.— ЭБС «IPRbooks»

**Дополнительная**

1. **Алехина, Л.Ф.** Английский язык. Фонетика. Морфология. Времена группы Indefinite [Электронный ресурс]: рабочий учебник/Алехина Л.Ф. - 2011. - http://lib.muh.ru

2. **Мамонова, Л.А.** Английский язык. Практика перевода. Часть 1 [Электронный ресурс]: рабочий учебник/Мамонова Л.А. - 2011. - http://lib.muh.ru

3. **Ромашкина, С.В.** Английский язык [Электронный ресурс]: учебное пособие/ Ромашкина С.В.— Электрон. текстовые данные.— Самара: РЕАВИЗ, 2010.— 70 c.—: http://www.iprbookshop.ru/10172.— ЭБС «IPRbooks»

**Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «АНГЛИЙСКИЙ ЯЗЫК. БАЗОВЫЙ КУРС ДЛЯ НЕЛИНГВИСТОВ»**

Ответственный за выпуск Е.Д. Кожевникова
Корректор И.А. Князева
Оператор компьютерной верстки В.Г. Буцкая

**4415.01.01;МУ.01;2**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**
**ПО ДИСЦИПЛИНЕ «НЕМЕЦКИЙ ЯЗЫК. БАЗОВЫЙ КУРС**
**ДЛЯ НЕЛИНГВИСТОВ» (КУРС 1)**

**МОСКВА 2018**

Разработано Л.Д.Захаровой, к.фил.н., доц.
Под ред. В.Н. Базылева, д.фил.н., проф.

Рекомендовано Учебно-методическим советом в
качестве методических указаний для обучающихся

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

# ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «НЕМЕЦКИЙ ЯЗЫК. БАЗОВЫЙ КУРС ДЛЯ НЕЛИНГВИСТОВ (КУРС 1)»

Методические указания подготовлены для обучающихся в образовательной организации и предназначены для овладения умениями речевого общения, использования и коррекции речи в профессиональных целях по направлению  в рамках дисциплины «Немецкий язык. Базовый курс для нелингвистов» (курс 1)

**ОБЩИЕ ПОЛОЖЕНИЯ**

**Цель практического занятия** заключается в формировании коммуникативной компетенции студента, позволяющей вступать в коммуникацию и уметь ориентироваться и реализовывать коммуникативные намерения в основных ситуациях общения (бытовых, социально-культурных, учебно-производственных); вступать в коммуникацию, задавать вопрос и сообщать о факте или событии, выражать намерение, желание, просьбу, пожелание и т.п.; выражать свое отношение к лицу, предмету, факту, событию.

**Задачи практического занятия:**

- формирование навыков и умений устной (монологической и диалогической) и письменной речи на немецком языке;

- активизация грамматических навыков;

- совершенствование лексических навыков, связанных с умением использовать лексико-фразеологические средства языка в речи, подбирать синонимические средства языка;

- совершенствование дискурсивных умений, связанных с оценкой типа текста, вариантов речи нормативного и ненормативного характера;

- формирование лингвокультурной компетенции, предполагающей знакомство с речевым этикетом, стереотипами речевого общения в немецкоязычной культуре;

- формирование стратегической компетенции, включающей в себя речевую активность, устойчивую потребность в общении на немецком языке;

- формирование у обучаемых заинтересованности в самообразовательной деятельности для более глубокого и осмысленного усвоения программных положений учебной дисциплины.

**Особенность** данного вида практических занятий заключается в коммуникативной направленности занятий, предполагающего последовательности осуществления практических и познавательных действий. Коммуникативные игры обладают высокой степенью наглядности и позволяют активизировать изучаемый языковой материал в речевых ситуациях, моделирующих и имитирующих реальный процесс общения.

Решение языковой задачи предусматривает формирование или совершенствование речевых навыков в процессе целенаправленного использования заданного языкового материала в речевой деятельности. Коммуникативная задача заключается в обмене информацией между участниками игры в процессе совместной деятельности.

Коммуникативная игра как особый вид занятия состоит из нескольких частей:

1) *вступительная*. Обучаемые знакомятся с темой, целью, порядком проведения занятия, его значимостью для профессиональной деятельности, критериями оценки качества отработки заданий, рекомендациями по использованию учебной литературы. На этом этапе осуществляется постановка конкретной задачи, моделирующей будущую профессиональную деятельность;

2) *подготовительная*. Обучаемые заняты коллективной работой: они продумывают речевые задачи каждой роли, формулируют реплики, обсуждают их уместность, языковую и речевую грамотность, интонационную реализацию;

3) *практическая*. Этап реализации речевых заданий: учащиеся произносят подготовленные диалоги и монологи, импровизируют на основе подготовленного речевого материала;

4) *заключительная*. Подведения итогов и контроля качества усвоения материала и оценки умений работы с текстами и лингвистическими базами на немецком языке: смысловой, лингвистический и стилистический анализ речевого материала. Подводятся итоги занятия, обучаемым выставляются оценки.

## Практическое занятие №1. ДЕЛОВАЯ ИГРА «ЗНАКОМСТВО», «Простое предложение»

***Цель работы***: *Усвоение речевых формул приветствия, прощания, представления, благодарности*

Для выполнения практического занятия обучающимся необходимы одноязычные словари немецкого языка (http://www.babla.ru/), карточки с персональными заданиями.

Игра 1. Каждый студент получает карточку, на которой написано имя-фамилия, профессия и возраст. Студенты знакомятся с образцом и каждый готовит о себе подобный рассказ.

Hallo! Ich heiße Felix Dietrich. Ich bin ledig. Ich bin schlank, sportlich, aktiv, nett und freundlich. Ich komme aus Deutschland, aus Hamburg. Ich lebe jetzt in München. In München studiere ich an der Uni. Ich studiere Jura. Ich spreche gut Englisch und natürlich Deutsch, das ist meine Muttersprache. Jetzt lerne ich Spanisch. Ich besuche zweimal pro Woche einen Spanischkurs. Aber mein Spanisch ist noch nicht sehr gut. In der Freizeit besuche ich meine Freunde, wir gehen oft ins Kino oder ins Cafe. Ich spiele Klavier und Gitarre, und ich höre Musik auch gern. Ich mache auch sehr gern Sport. Ich spiele Tennis und Fußball, und ich gehe auch schwimmen. Und ich reise sehr gern. Am Abend sitze ich oft am Computer und lerne für die Uni, surfe im Internet oder spiele.

*2. Игра 2. Диалоги. Студенты делятся на группы по 3, и каждый готовит представление 2 студентов на основе предложенных образцов*

- Hallo! Sagen Sie mir bitte, wie spät es ist?
- Es ist halb acht.
- Danke schön, ich habe noch so viel Zeit! Zum Glück ist das Wetter sehr schön. Wenn es regnete, wäre es nicht so angenehm.
- Na ja, ich bin mit Ihnen einverstanden. Ich habe ein Treffen mit den Freunden um neun Uhr vereinbart, jetzt muss ich auch auf sie warten.
- Wirklich? So ein Zufall! Ich auch. Meine Mutter sagt immer: Katya, du solltest dich öfter erholen! Aber ich arbeite viel und kann mir das nicht leisten.
- Noch ein Zufall! Mein Name ist auch Katya!
- Na ja, das klappt nicht sehr offen.
- Und wo arbeiten Sie?
- Im Geschäft. Ich besitze ein Lebensmittelgeschäft und arbeite auch dort.
- Das ist der Grund, warum Ihr Gesicht mir so bekannt vorkommt. Ich kaufe Lebensmittel in Ihrem Geschäft jeden Tag.
- Merkwürdig, dass wir uns hier begegnet haben. Ich bin in diesem Stadtteil zum ersten Mal!
- Ich auch. Freut mich, Sie kennenzulernen.
- Gegenseits!

***

- Hallo, machen wir uns bekannt.
- Hallo, ich heiße… Ich bin … Jahre alt. Und Sie?
- Sie dürfen mich duzen. Mein Name ist… und ich bin … Jahre alt.
- Sehr angenehm! Ich komme aus… Und woher kommst du?
- Ich komme aus… Was bist du? Bist du Student?
- Nein, ich gehe noch in die Schule. Und du?
- Ich auch! Liebst du deine Schule?
- Selbstverständlich! Mein Lieblingsfach ist…
- Ich liebe … auch! Wir haben viel gemeinsames!
- Kannst du mir bitte deine Telefonnummer geben?
- Warum nicht?

***

- Darf ich mich vorstellen: Weber, Hans Weber. Ich komme aus Dortmund.
- Sehr angenehm. Mein Name ist Erika Koch. Ich komme aus Köln.

***

- Das ist meine Familie: meine Frau und meine Kinder. Das ist mein Sohn. Er heißt Peter. Und hier sind meine Töchter.
- Wie heißen Ihre Töchter?
- Sie heißen Sabine und Susanne.

***

- Das ist mein Mann und das sind meine Kinder. Das hier ist meine Tochter. Sie heißt Birgit. Und das sind meine Söhne. Der ältere heißt Mattias und der jüngere Franz.

- Das sind meine Eltern und meine Brüder.
- Und wer ist das?
- Das ist mein alterster Brüder Heinrich.

*1. Игра 3. Студенты делятся на группы, каждая группа получает карточку со словами. Из этих слов студенты должны составить как можно больше предложений на немецком языке. Предложения из одних и тех же слов, но с различным порядком слов засчитываются как разные. Затем каждая из групп презентует свои предложения, а задача других команд – составить предложения со словами другой команды, такие, которые отсутствуют в презентации. Побеждает та команда, которая предложила как можно большее количество предложений, и со словами которой конкуренты составили наименьшее количество предложений*

**Карточки со словами**

**№ 1**

| |
|---|
| essen (aß, gegessen) |
| schmecken: das schmeckt |
| trinken (trank, getrunken) |
| das Brot, -e |
| das Ei, -er |
| die Suppe, -n |
| sauer |
| süß |
| warm |

**№ 2**

| |
|---|
| kochen |
| möchten |
| glauben |
| das Hähnchen |
| das Fleisch |
| der Fisch, -e |
| alt |
| hart |
| bitter |

**№ 3**

| |
|---|
| der Teller |
| das Messer |
| der Löffel |
| trocken |
| salzig |
| scharf |
| bringen (brachte, gebracht) |
| einkaufen |

**№ 4**

| |
|---|
| die Gabel, -n |
| das Glas, -er |
| die Flasche, -n |
| mittags |
| abends |
| manchmal |
| putzen |
| spülen |
| waschen |

**№5**

| |
|---|
| der Wein, -e |
| das Bier, -e |
| der Saft, -e |
| genug |
| satt |
| hungrig |
| brauchen |
| nehmen |
| bezahlen |

**Практическое занятие №2. КОММУННИКАТИВНАЯ ИГРА «мОЙ ДОМ», «Род и число существительного»**

***Цель работы***: *Усвоение речевых формул, лексики по теме: «моя семья», «мой дом»; формирование умений монологического описания, а также обиходно-бытового диалога; совершенствование грамматических умений использования имени существительного*

Для выполнения практического занятия обучающимся необходимы тексты и словари для анализа (http://www.babla.ru/), фотография (http://www.xbox-passion.de/attachments/f13/17325d1435039735-psydee-hat-geburtstag-alles-gute-geburtstag-635217088976590000.jpg).

*Игра 2. Каждый студент готовит схематический план своей квартиры, одновременно студент готовит вопросы к собеседнику о его квартире. Далее студенты разбиваются на группы, один отвечает на вопросы о квартире (доме), другие – задают вопросы, они пользуются и приведенными ниже диалогами.*

Barbara: Ich habe gehört, dass deine Familie eine wunderbare Wohnung gemietet hat!

Irmtraut: Die Wohnung und deren Lage sind wirklich wunderbar. Aber wir haben sie nicht gemietet. Wir haben sie gekauft. Jetzt ist das unsere Eigentumswohnung.

Barbara: Echt? Soll das ein Witz sein?

Irmtraut: Es ist kein Spaß. Das ist die pure Wahrheit.

Barbara: Gratulationen!

Irmtraut: Danke, wir sind richtig glücklich. Ich würde dir unsere Wohnung gerne zeigen.

Barbara: Vielleicht gehen wir gleich hin? Was hast du heute noch vor? Ich bin so gespannt!

Irmtraut: Gehen wir! Ich habe nichts Besonderes vor.

Barbara: Klasse!

In der Wohnung:

Irmtraut: Hier haben wir eine traumhafte Diele. Hier gab es genug Platz für eine große Garderobe, für ein Sofa und sogar für ein paar schöne Palmen.

Barbara: Du hast alles sehr schön eingerichtet!

Irmtraut: Das war die Idee von meinem Mann. Holger wollte immer ein Sofa in der Diele haben. Und Palmen sind seine Lieblingspflanzen, wie du bestimmt weißt.

Barbara: Ja, diese Liebe von Holger ist allen bekannt.

Irmtraut: Die linke Tür führt aus der Diele in unser Schlafzimmer, die rechte - in das Arbeitszimmer meines Mannes. Und hier ist der Eingang in unser Wohnzimmer.

Barbara: Hier gibt es aber noch Türen.

Irmtraut: Drüben sind Badezimmer, Toilette und kleiner Abstellraum.

Barbara: Und wo hat sich eure Küche versteckt?

Irmtraut: In die Küche geraten wir direkt aus unserem Wohnzimmer. Ich finde das sehr bequem.

Barbara: Das ist natürlich eine Gewohnheitssache.

Irmtraut: Hier ist unser Schlafzimmer. Es ist noch nicht eingerichtet, weil wir kein passendes Bett gefunden haben. Hier haben wir auch einen großen Balkon.

Barbara: Dein Balkon ist großartig! Und die Aussicht ist auch schön. Zeige mir bitte deine Küche, ich bin gespannt, wo du jetzt kochst.

Irmtraut: Das ist unsere neue Küche. Hier verbringe ich viel Zeit. Hier gibt es alles, was man zum Kochen brauchen kann.

Barbara: Ja, du hast einen sehr modernen Gasherd. Deine Mikrowelle ist auch super. Und so viele Küchenhelfer!

Irmtraut: Die Küche ist sehr wichtig für uns. Kochen macht uns Spaß und wir haben oft Gäste.

Barbara: Wann lädtst du mich ein?

Irmtraut: Das ist eine gute Frage. Bald organisieren wir natürlich eine Einzugsfeier. Ich weiß jetzt nicht genau, wann wir das machen. Hoffentlich wird unsere Wohnung in zwei-drei Wochen vollständig eingerichtet. Jedenfalls melde ich mich bei Dir und sage Bescheid.

Barbara: Danke für die Einladung! Ich warte schon auf eure Einzugsfeier mit großer Ungeduld.

Irmtraut: Ich auch.

*Игра 2. Индивидуальная игра-эстафета. Перед студентами предложения. Каждый по очереди определяет форму существительного и описывает его форму*

1. Ich habe einen Bruder und eine Schwester. Mein Bruder ist 11 Jahre alt und die Schwester ist 22. 2. Die große russische Stadt St. Petersburg liegt an der Newa. 3. Kurt wohnt in einer schönen Stadt im Süden der Bundesrepublik Deutschland. 4. In unserer Klasse stehen neun Schulbänke und ein Lehrertisch. 5. Links auf dem Tisch liegen Schulsachen. Das sind Annas Schulsachen. Ihre Schulsachen sind wie immer in Ordnung. 6. Ich trinke sehr gern Tee und mein Vater - Bier. 7. Das Wetter ist wunderschön heute - die Sonne scheint, der Himmel ist blau und es gibt dort keine Wolken. 8. Der Deutschlehrer kommt in die Klasse und die Stunde beginnt. 9. Seine Mutter ist Krankenschwester, die Frau arbeitet in einem Krankenhaus im Zentrum der Stadt. 10. Das Drama des großen deutschen Dichters Johann Wolfgang von Goethe „Faust" ist weltbekannt. 11. Ich schreibe heute einen Brief an meine Freundin. 12. Dieser Ring ist sehr teuer, er ist aus Gold. 13. Der Vater meines Freundes fährt im Herbst nach Deutschland. 14. In der Literaturstunde schreiben die Schüler am Dienstag ein Diktat. 15. Brot, Milch, Fleisch, Butter, Fisch sind Lebensmittel und Limo, Kaffee, Cola, Bier sind Getränke. 16. Die Stadt ist nicht groß, aber es gibt hier ein Theater, drei Kinos, ein Kunstmuseum und eine Gemäldegalerie. 17. Touristen aus aller Welt besuchen gern die Dresdener Gemäldegalerie und bewundern ihre Kunstschätze. 18. Ich treibe gern Sport, aber nicht immer habe ich dafür Zeit. 19. Montag ist der erste Tag der Woche. 20. Im Cafe bestellen wir Kaffee. Der Kaffee schmeckt sehr gut. 21. Die Katze ist ein Haustier. 22. Der Herr dort links ist unser Professor. 23. Herr Müller ist Deutsche. 24. Die größte Stadt Deutschlands ist Berlin, es hat 3,5 Millionen Einwohner. 25. „Fräulein Eckardt! Zeigen Sie mir bitte diese Akten!" 26. Die größten Flüsse der Bundesrepublik Deutschland sind der Rhein, die Oder, die Elbe, der Main und die Donau. 27. Olgas Onkel ist Arzt von Beruf. Er ist ein guter Arzt. 28. Das ist ein Geschenk von meinem Großvater. 29. Und dieses Buch ist das Geburtstagsgeschenk meiner Mutter. 30. Kein Mädchen in der Klasse singt so schön wie Monika. 31. Nicht jeder Schüler unserer Klasse interessiert sich für Chemie.

**Практическое занятие № 3. КОММУНИКАТИВНАЯ ИГРА «ГЛАГОЛ», «РАСПОРЯДОК ДНЯ»**

***Цель работы***: *Усвоение речевых формул, лексики по теме: «распорядок дня»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на группы (команды), в каждой не менее 3-4 и не более 6 человек. Каждая группа получает задание подготовить презентацию. Преподаватель ведет игру. Он называет глаголы. Каждая команда за отведенные 3 минуты должна составить несколько предложений с этим глаголом. Побеждает та команда, которая составила максимальное количество правильных предложений.*

1. wiederholen; 2. untersuchen; 3. stattfinden; 4. fernsehen; 5. widersprechen; 6. festlegen; 7. teilnehmen; 8. festsetzen; 9. zurückkommen; 10. festnehmen; 11. freilassen; 12. zurückgeben; 13. feststellen; 14. zurückkehren; 15. freisprechen; 16. rechtfertigen; 17. zurücknehmen; 18. hochheben; 19. zurücklegen. 20. zurückgeben; 21. fernsprechen; 22. teilnehmen; 23. festhalten; 24. vorbeifahren; 25. lobpreisen; 26. festsetzen; 27. freisprechen; 28. gewährleisten. 29. freilassen; 30. weggehen.

*Игра 2. Каждый студент готовит рассказ о своем распорядке дня, одновременно студент готовит вопросы к собеседнику о его обычном дне. Далее студенты разбиваются на группы, один отвечает на вопросы о распорядке дня, другие – задают вопросы, они пользуются и приведенными ниже диалогами и текстами.*

A: Also, Herr Krause, was haben Sie gestern gemacht?

B: Gestern, Herr Vorsitzender, habe ich nichts gemacht.

A: Nun, irgendwas haben Sie doch gemacht.

B: Nein, Herr Vorsitzender, ganz bestimmt nicht.

A: Einen Spaziergang, zum Beispiel. Haben Sie nicht wenigstens einen Spaziergang gemacht?

B: Nein, Herr Vorsitzender, ich habe gestern keinen Spaziergang gemacht.

A: Nun, denken Sie mal ein bisschen nach, Herr Krause…

B: Das tue ich ja, Herr Vorsitzender, ich denke schon die ganze Zeit nach.

A: Aha, Sie denken schon die ganze Zeit nach. Wie lange denn schon?

B: Ich weiß nicht… ich denke viel nach, immer wieder denke ich nach.

A: Haben Sie vielleicht gestern auch nachgedacht?

B: Ich glaube ja, Herr Vorsitzender.

A: Na sehen Sie! Sie haben gestern also doch etwas gemacht!

B: Na ja, das heißt…

A: Haben Sie gestern nachgedacht, ja oder nein?

B: Ja.

A: Na also!

B: Ist das verboten?

A: Herr Krause - hier stelle ich die Fragen!

B: Entschuldigung.

A: Sie können gehen!

***

- Wie spät ist es, Lina? Meine Uhr steht.

- Es ist ein Viertel nach zehn.

- So spät? Geht deine Uhr nicht vor?

- Nein, sie geht richtig.

- Dann muss ich mich beeilen. Ich muss punkt 12 im Institut sein. Wir haben eine Konsultation.

- Anna, da hast du noch viel Zeit bis dahin.

- Ich muss noch unser Zimmer aufräumen, Geschirr abwaschen, Brot und Käse holen. Diese Woche habe ich in unserer Familie Dienst. Außerdem muss ich noch mein Kleid bügeln.

- Die Einkäufe kannst du unterwegs machen.

- Das sowieso.

- Hast du heute abends etwas vor.

- Nein. Ich habe nichts vor. Ich komme gegen 6 zurück und werde Musik hören. Ich schwärme für Mozart und für die Oper „Barbier von Sevilla“.

- Weiß du, ich komme dann vielleicht zu dir. Natürlich, wenn nichts dazwischenkommt.

- Ja, komm bitte. Ich bin heute Abend ganz allein, meine Eltern machen heute einen Besuch.

- Abgemacht. Ich komme zu dir gegen 20 Uhr. Ist es nicht zu spät?

- Doch, es ist zu spät. Die Oper dauert etwa 3 Stunden.

- Vielleicht verspäte ich mich ein wenig.

- Dann komm, wann es dir recht ist.

***

Freizeit bedeutet für alle Menschen Entspannen, Abschalten von Problemen. Das heißt aber nicht, dass sie total faulenzen mögen. Im Gegenteil, sie bevorzugen etwas Interessantes zu machen, was ihnen gefällt, und nicht, was man machen soll. Man kann seinen Interessen nachgehen, Abenteuer erleben, die man sich wegen Zeitnot nicht leisten kann.

Fast jeder Mensch hat ein oder viele Hobbys. Meine liebste Freizeitbeschäftigung ist Lesen. Ich lese Bücher von Kindheit an. Mit 12 Jahren begann ich ernste Bücher von Dickens, Duma, Tolstoi, Dostojewski zu lesen. Ich las jedes Buch nicht nur einmal, ich las es mehrere Male. Wie spannend wurde dort die Welt geschildert!

In den Ferien haben wir gewöhnlich mehr Freizeit. Spaziergehen, Bücher lesen, Sport treiben, Wandern – das Angebot kennt keine Grenzen wie unsere Phantasie. Was die Jugendlichen betrifft, so ziehen sie heutzutage am liebsten vor, im Internet stundenlang zu surfen oder Computerspiele zu spielen. Was mich angeht, so mache ich Vieles gern. Ich lese, sehe fern, treffe mich mit Freunden, gehe ins Theater und Kino und treibe auch Sport mit Vergnügen. Ich will mich weiterbilden, wenn es um interessante Dinge geht.

Ich reise sehr gern und jede Reise bringt viele Eindrücke. Es ist spannend und interessant, neue Landschaften zu sehen und neue Bekanntschaften zu machen. Zu meinem schönsten Erlebnis aus den vorigen Sommerferien zählt die Wanderung aufs Land. Es war so schön für einige Tage ins Grüne zu fahren, in der Sonne zu liegen, im See zu baden, Pilze und Beeren im Wald zu sammeln. Meine Freunde angelten gern. Wir nahmen das Allernötigste mit und transportierten unser Gepäck mit unseren Fahrrädern. Wir schlugen das Zelt auf und machten das Feuer an. Wenn es schon dunkel wurde, saßen wir um das Feuer, spielten Gitarre und sangen. Es war so schön ein bisschen weit von der Zivilisation zu hausen.

## Практическое занятие № 4. КОММУНИКАТИВНАЯ ИГРА «МОДАЛЬНЫЕ ГЛАГОЛЫ», «ДЕНЬ РОЖДЕНИЯ»

***Цель работы***: *Усвоение речевых формул, лексики по теме: «день рождения», «вечеринка»; речевых формул поздравления и пожелания; формирование умений монологического описания, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты и словари для анализа (http://www.babla.ru/), фотография (http://www.xbox-passion.de/attachments/f13/17325d1435039735-psydee-hat-geburtstag-alles-gute-geburtstag-635217088976590000.jpg).

*Игра 1. Викторина. Задача студентов вставить пропущенный модальный глагол в предложение. Ответивший правильно получает фишку, победителем становится тот, у кого много фишек.*

1. Leider _____ich nicht länger bei dir bleiben, denn ich _____um 17 Uhr mit dem Zug nach München fahren.

2. Eis oder Kaffee? Was _____ du?

3. Ich _____ keinen Kaffee trinken; der Arzt hat’s mir verboten.

4. Ich _____ täglich dreimal eine von diesen Tabletten nehmen.

5. Wo _____ du denn hin? _____ du nicht einen Moment warten, dann gehe ich gleich mit dir?

6. “Guten Tag! Wir _____ einen Doppelzimmer mit Bad; aber nicht eins zur Straße. Es _____ also ein ruhiges Zimmer sein.” - “Ich _____Ihnen ein Zimmer zum Innenhof geben. _____ Sie es sehen?” -”Ja, sehr gern”- “_____ wir Sie morgen früh wecken?” - Nein, danke, wir _____ausschlafen.”

7. Herr Müller _____ ein Haus bauen.

8. Er _____ lange sparen.

9. Auf den Kauf eines Grundstücks _____ er verzichten, denn das hat er schon.

10. Er _____ laut Vorschrift einstöckig bauen.

11. Den Bauplan _____ er nicht selbst machen. Deshalb beauftragt er einen Architekten; dieser _____ ihm einen Plan für einen Bungalow machen.

12. Der Architekt _____ nur 1500 Mark dafür haben; ein “Freundschaftspreis”, sagt er.

13. Einen Teil der Baukosten _____ der Vater finanzieren. Trotzdem _____ sich Herr Müller noch einen Kredit besorgen.

14. Er _____ zu den Banken, zu den Ämtern und zum Notar laufen. - Endlich _____ er anfangen.

15. Der Mann hat doch eine Verletzung! Wer das nicht sieht, ..... blind sein.

16. Du ..... recht haben; aber es klingt sehr merkwürdig.

17. Diese Schauspielerin ..... 80 Jahre alt sein, so steht es in der Zeitung. Sie sieht doch aus wie fünfzig!

18. Der Junge ..... die Geldbörse gefunden haben; dabei habe ich gesehen, wie er sie einer Frau aus der Einkaufstasche nahm.

19. “Er ..... ein Vermögen von zwei bis drei Millionen besitzen, glaubst du das?” - “Also das ..... übertrieben sein. Es ..... sein, dass er sehr reich ist, aber so reich sicher nicht!”

20. In Griechenland ..... gestern wieder ein starkes Erdbeben gewesen sein.

21. Es ist schon zehn Uhr. Der Briefträger ..... eigentlich schon dagewesen sein.

22. Eben haben sie einen Fernsehbericht über Persien angekündigt, jetzt zeigen sie Bilder über Polen. Da ..... doch wieder ein Irrtum passiert sein!

23. Wir haben dein Portemonnaie in der Wohnung nicht gefunden. Du ..... es nur unterwegs verloren haben. Wenn du es nicht verloren hast, ..... es dir gestohlen worden sein.

24. Den Ring ..... sie geschenkt bekommen haben, aber das glaube ich nicht.

25. Er ist erst vor zehn Minuten weggegangen. Er ..... eigentlich noch nicht im Büro sein.

26. Es ..... heute Nacht sehr kalt gewesen sein, die Straßen sind ganz vereist.

27. _____ du schwimmen?

28. Der Verletzte _____ schon laufen.

29. Er _____ mir helfen, aber er will nicht.

30. Du _____ das nicht (tun).

31. Der Angeklagte _____ sofort zum Richter kommen?

32. _____ ich diesen Fall allein aufklären?

33. Alle Kinder _____ zur Schule gehen.

34. Sie _____ bei uns noch ein paar Tage bleiben.

35. Ich _____ mit ein paar Worten den Vorfall beschreiben.

36. Das _____ niemand erfahren.

37. Niemand _____ die Gesetze verletzen.

38. Ohne Zeugen _____ man die Haussuchung nicht machen (halten).

39. Der Zeuge _____ Aussagen machen.

40. Du _____ nicht so viel rauchen.

*Игра 2. Каждый студент готовит сообщение о своем дне рождения, одновременно студент готовит вопросы к собеседнику. Далее студенты разбиваются на группы, один отвечает на вопросы, другие – задают вопросы, они пользуются и приведенными ниже диалогами.*

Meinen Geburtstag feiere ich gewöhnlich zu Hause. Am 2. Dezember laden wir keine Gaste ein, weil meine Mutti sagt: "Es ist zu teuer" und ich denke - es ist viel angenehmer meinen Geburtstag im Familienkreis zu feiern. Üblich

backt meine Mutter eine Torte fur den Feiertagstisch selbst, aber wenn sie keine Lust dazu hat, kaufen wir die Torte im Geschäft "...", das in der Nähe von unserem Haus liegt. Aber auf jeden Fall gefallt mir die Torte meiner Mutter viel besser. Von Morgen an bereiten wir (ich und meine Mutti) verschiedene Speisen (Salaten, Getränke). Ungefähr um 16 Uhr, wenn wir schon ganz fertig sind, laden wir unsere Oma und unseren Opa, meinen Cousin und meine Cousine, andere Verwandten ein. Danach, wenn sie noch unterwegs sind, decken wir den Tisch. Nach dem an angenehmsten Teil meines Geburtstags (Geschenken) setzen wir meinen Bruder vom Computertisch zum Esstisch um und beginnen reden, essen und Trinkspruche in meiner Ehren ausbringen.

***

Ich bin 21 Jahre alt. Ich habe Geburtstag am 17. March und feiere ihn jedes Jahr. Ich feiere diesen Tag mit meiner Familie und manchmal lade viele Gaste ein. Sie sind meine Schul- und Institutsfreunde. Die Gaste kommen gewöhnlich Sonntags um 3 Uhr.

Wenn mein Geburtstag an einem anderen Tag der Woche ist, feiere ich nur mit meiner Familie und meinen Verwandten und die Gaste kommen am Wochenende.

Am Geburtstag erhalte ich Glückwunschkarten und Geschenke. Ich bekomme Bucher, Tonbande, Blumen, Parfumerie und andere Sachen. Meine Freunde rufen mich auch an, um mir zu gratulieren.

Vor meinem Geburtstag raume ich die Wohnung auf und kaufe Lebensmittel und Getränke. Am Morgen decke ich den festlichen Tisch, bringe alles in Ordnung und warte auf die Gäste.

Wenn die Gäste kommen, hören wir Musik, essen, trinken Sekt und Wein und tanzen. Am Abend essen wir auch den Geburtstagkuchen mit Tee. Das festliche Essen bereitet meine Mutter zu, und ich und meine Großmutter backen einen Geburtstagskuchen.

Wir verbringen die Zeit lustig und feiern gewöhnlich bis 11 Uhr.

***

Ende Mai hat mein Bruder Peter seinen Geburtstag. Er möchte eine große Party für seine Bekannte und Freunde organisieren. Er hat entschieden, ca. 20 Personen einzuladen. Diese Party wird in seinem Sommerhaus am letzten Maiwochenende stattfinden.

Peter wird für das Essen selbst sorgen. Er möchte Orangen, Äpfel, Weintrauben, Birnen, Erdbeeren, Pfirsiche, Bananen und Kiwi kaufen. Obst muss nur gewaschen werden, danach wird es einfach in großen Schalen serviert. Was Gemüse anbetrifft, so hat er sich für Gurken, Tomaten, Paprika und verschiedene Kräuter entschieden. Natürlich wird eine ausreichende Menge Weiß- und Schwarzbrot besorgt.

Mein Bruder hat eine sehr schöne große Gartenlaube, wo er Barbecue machen möchte. Hier wird uns kein Regen stören, obwohl wir alle hoffen, dass das Wetter schön sein wird. Peter kauft zehn Kilo Schweinefleisch und ich werde das Fleisch für Barbecue vorbereiten. Ich werde es in Portionsstücke schneiden und am Vorabend einlegen. Das kann ich sehr gut machen, weil ich viele gute Rezepte kenne.

Nach dem Essen wird es getanzt, Karaoke gesungen und gespielt. Ich bin sicher, es wird eine unvergessliche Party sein!

*Игра 3. Каждый студент готовит письменное поздравление с днем рождения для а) своего друга, б) одного из своих родителей, в) одного из своих близких родственников. Задание каждый получает в результате жеребьевки, затем каждое предложение зачитывается вслух.*

Ich gratuliere! Und ich freu' mich sehr, denn dieser Glückwunsch fällt mir gar nicht schwer: Erhalten bleibe stets der alte Schwung! Dann macht das Leben Spaß; dann bleibt man jung!

Man wird nicht älter, sondern besser! Happy Birthday!

Bleibe fröhlich, frisch und munter, wie ein Fisch und geh nicht unter. Nach einem Glückwunsch ist mir sehr zumute, ich gratuliere herzlich: Alles Gute!

Du musst echt was ganz besonderes sein! Heute haben 4.534.567 Leute Geburtstag, aber ich habe nur an Dich gedacht! Alles Gute und Liebe!

Ein Engel schaut von oben runter, holt schnell die Sterne, die singen munter ein Geburtstagsliedchen nur für Dich, einfach so, weil es Dich gibt! Alles Gute!

Ich wünsche dir zum Wiegenfeste von ganzem Herzen alles Beste und außerdem das ist ganz klar! Ein schönes neues Lebensjahr.

Jahre sind vorbei, nicht alle waren sorgenfrei. dein neues Lebensjahr sei heiter, das Schönste auf der Lebensleiter!

Heitere Tage, frohe Stunden, viel Erfolg mit Glück verbunden, stets Gesundheit, Sinn zum Scherzen dieser Wunsch heut' kommt vom Herz.

Liebe Glückwünsche für den heutigen Tag und alles Gute für die Zukunft wünscht…

Mögen alle deine Wünsche in Erfüllung gehen liebe … Herzliche Geburtstagsgrüße

Alles Glück dieser Erde soll dein ständiger Begleiter sein. Deine Wünsche und Träume sollen in Erfüllung gehen. Herzliche Glückwünsche zum Geburtstag.

Ich bin ein kleiner Pinkel, rund und dick: Ich schlüpfe aus dem Winkel und wünsche dir viel Glück, Alles Liebe zum Geburtstag

Ich wünsche dir mit Hand und Mund und aus tiefsten Herzensgrund: Sei glücklich, immer dar im neu begonnenen Lebensjahr. Alles Liebe zum Geburtstag …

Jahre sind es wert, Dass man Dich besonders ehrt. Darum wollen wir Dir heut' sagen, Es ist schön, dass wir Dich haben!

## Практическое занятие № 5. РОЛЕВАЯ ИГРА «В МАГАЗИНЕ», «ИНФИНИТИВНЫЕ КОНСТРУКЦИИ»

***Цель работы***: *Усвоение речевых формул, лексики по теме: «магазин», «продукты», «одежда»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание: «Купить продукты для праздника», «Купить продукты на неделю», «Купить подарок другу на день рождение», «Купить новую одежду (обувь)», «Купить мебель (бытовую технику)», «Купить книгу» и под. Далее каждая группа представляет свой диалог.*

1)

- Olga: Hallo, Anna! Heute habe ich dich extra eingeladen, mir zu helfen. In einer Woche fahren wir für ein paar Wochen in unser Sommerhaus und ich möchte in diesem Jahr das neue Geschirr kaufen, weil wir wie immer viele Gäste einladen werden.
- Anna: OK, ich helfe dir gerne. Hast du dir schon überlegt, wo wir das Geschirr kaufen werden?
- Olga: Ja, ich möchte unbedingt zu IKEA fahren. Und danach vielleicht noch in ein großes Kaufhaus.
- Anna: OK, fahren wir zuerst zu IKEA, ich kaufe dort auch gerne ein. Die Waren sind dort preiswert und qualitätsgerecht.
- Olga: Zuerst schauen wir uns Töpfe und Pfannen an. Ich möchte einen großen Kochtopf aus rostfreiem Stahl mit dickem Boden kaufen. Und eine neue Pfanne brauchen wir auch.
- Anna: Brauchst du auch neue Deckel?
- Olga: Nein, wir haben genug Deckel, die auch für diese Kochtöpfe passen.
- Anna: Wie schön sind diese bunte Schüssel da! Die möchte ich unbedingt haben. Und ich kaufe auch so ein Geschenk für meine Mutter. Sie mag solche Sachen.
- Olga: Und ich werde sie nicht kaufen. Sie gefallen mir gut, aber ich habe viele ähnliche Sachen auf dem Lande. Was ich aber wirklich brauche, sind kleine und große flache Teller und große Tassen.
- Anna: Wie findest du dieses bunte Set? Meiner Meinung nach ist er einfach Klasse! Und es gibt auch passende Tassen in der von dir erwünschten Größe!
- Olga: Du hast Recht, ich glaube das ist genau das, was meine Familie braucht. Wir nehmen dieses Set und gehen dann an die Kasse. Ich meine, dass das Allerwichtigste wir heute besorgt haben.

***

2)

- Heute möchte ich gerne einkaufen. Ich habe genug Zeit, und morgen fährt unsere Familie aufs Land. Ich möchte alles für diese Reise besorgen.

- Morgen fahren wir auch ins Grüne. Und mein Mann hat mich gebeten, entsprechende Lebensmittel zu kaufen.

- Vielleicht gehen wir zusammen? Es gibt eine große Kaufhalle in der Nähe. Dort finden wir viele gute Waren und Lebensmittel.

- Gerne! Zusammen ist immer besser für mich. Oft brauche ich einen Rat.

- Dann gehen wir.

**In der Kaufhalle**

- Oh, das ist eine wirklich große Kaufhalle! Hier bin ich noch nicht gewesen. Ich bin gespannt, was da alles verkauft wird.

- Hier kann man alles kaufen, was man braucht. Meine Mutter hat mir diese Kaufhalle vor ein paar Monaten angeraten.

- Also, gehen wir zuerst in die Abteilung, wo frisches Gemüse und Obst verkauft werden.

- OK, einverstanden.

- Guck mal, die Auswahl ist super! Und die Qualität der Produkte scheint sehr gut zu sein. Diese Tomaten, Gurken und Salatzwiebeln gefallen mir sehr gut. Und es gibt alle Kräuter, die ich brauche – Dill, Petersilie, Basilikum, Koriandergrün… Ich kaufe noch Eisbergsalat und Schnittlauch.

- Ich kaufe auch einige Kräuter, Porree, Paprika, Gurken, Tomaten und Zwiebeln. Ich habe zu Hause leckeren Schafkäse und möchte morgen den «griechischen Salat» machen.

- Oh, ich liebe diesen Salat! Den bereite ich im Sommer auch oft zu. Insbesondere wenn wir etwas grillen.

- Morgen macht mein Mann sein Firmenbarbecue. Zu diesem Gericht essen wir gerne den «griechischen Salat».

- Alles klar. Und ich sehe schon die Äpfel, die ich unbedingt kaufen werde. Diese Weintrauben sehen auch hervorragend aus.

- Obst brauche ich nicht, weil mein Mann es besorgt.

- Dann gehen wir weiter. Drüben ist eine riesengroße Fleischabteilung.

- Ich kaufe heute Putenfilet. Es ist immer ganz mager und sehr gesund.

- Putenfilet haben wir am letzten Wochenende gegrillt. Heute kaufe ich was anderes. Zum Beispiel, diese Schweinemedaillons. Mein Mann isst Schweinefleisch sehr gern.

- Was brauchen wir noch?

- Ich habe alles besorgt. Frische Brötchen kaufen wir natürlich morgen.

- Und ich brauche noch Getränke – Mineralwasser, Rotwein und Saft.

- Oh, über die Getränke habe ich völlig vergessen. Ich nehme sechs Flaschen Bier. Mineralwasser kaufen wir üblicherweise nicht, weil wir einen Brunnen neben unserem Landhaus haben.

- Ihr habt aber Glück!

- So ist das Leben.

- Wenn wir alles besorgt haben, gehen wir zum linken Ausgang. Daneben ist unsere Bushaltestelle.

- Ich danke dir recht herzlich für deine heutige Unterstützung.

- Macht nichts, das hat mir auch Spaß gemacht.

***

3)

Verkäuferin: Einen schönen guten Tag, kann ich vielleicht Ihnen helfen?

Der Kunde: Guten Tag, ja, bitte, Ihre Hilfe würde ich gerne gebrauchen.

Verkäuferin: Ist mir ein Vergnügen. Ich sehe, Sie suchen einen festlichen Anzug.

Der Kunde: Ja, stimmt, mein Bruder heiratet nächste Woche, ich bin dazu eingeladen. Mein alter Anzug ist mir leider zu eng geworden, deshalb brauche ich einen neuen.

Verkäuferin: Ist ja kein großes Problem. Wir suchen was ganz besonderes aus.

Der Kunde: Nein-nein, danke, gerade das möchte ich nicht. Ich bevorzuge eher unauffällige Anzüge.

Verkäuferin: Ach so, alles klar. In diesem Fall brauchen wir etwas klassisches, richtig?

Der Kunde: Ja, Sie haben vollkommen Recht. Klassik ist unsterblich und immer aktuell. Es würde mir passen.

Verkäuferin: Ok. Ich schätze, Sie haben die Größe L im Oberteil, aber Unterteil ist etwas kleiner, ich denke Größe M.

Der Kunde: Ja-ja, meine Figur ist ein bisschen unregelmäßig.

Verkäuferin: Keine Sorge, ich finde schon was. Welche Anzugsfarbe hätten Sie lieber: schwarz oder grau?

Der Kunde: Schwarz, ich mag schwarz.

Verkäuferin: Gut, dann probieren Sie bitte diesen Anzug an. Die Umkleidekabine finden Sie hinter den Tresen.

Der Kunde: Gut. Aber entschuldigen Sie bitte, die Hosenkanten hier unten sind nicht bearbeitet. Wie ist es möglich?

Verkäuferin: Ach das. Das ist aber kein Fehler, alles gehört dazu. Die Hosenkanten lassen wir absichtlich unbearbeitet, damit man die Hosenlänge korrigieren kann. Wenn der Anzug Ihnen passt, bearbeiten wir sofort hier auf der Stelle die Hosenkanten ihrer Größe nach. Alles klar?

Der Kunde: Ja, ich habe alles kapiert. Prima!

Verkäuferin: Na, wie geht es Ihnen? Wie sieht es aus?

Der Kunde: Mir gefällt es. Es passt ganz gut, aber die Ärmer sind etwas länger als es nötig ist.

Verkäuferin: Die bringen wir schon in Ordnung und machen sie gerne für Sie kürzer. Ja, dieser Anzug steht Ihnen wirklich sehr gut.

Der Kunde: Echt? Dann nehme ich ihn. Wie lange dauert es, die Hosenkanten und die Ärmel kürzer machen?

Verkäuferin: In einer Stunde soll es schon fertig sein. Aber wenn Sie keine Zeit zum Warten haben, können Sie uns Ihre Adresse hinterlassen. Wir schicken den fertigen Anzug für Sie nach Hause.

Der Kunde: Das wäre ja super. Was kostet er?

Verkäuferin: 800 Euro bitte.

Der Kunde: Ok, kann ich mit Kreditkarte bezahlen?

Verkäuferin: Ja, natürlich.

Der Kunde: Danke sehr. Einen schönen Tag noch.

Verkäuferin: Ebenso danke. Auf Wiedersehen.

***

4)

- Hallo, kann ich Ihnen behilflich sein?

- Hallo! Ich glaube, ja. Ich brauche eine festliche Bluse, und Ihre Auswahl an Blusen ist so groß, dass ich alleine zu viel Zeit verlieren werde.

- Es wird mir ein Vergnügen, ihnen zu helfen. Ich kenne mich hier sehr gut aus und wir werden schnell finden, was Sie brauchen. Erzählen Sie mir bitte ganz kurz über Ihre Wünsche und Vorstellungen.

- In zwei Wochen fliege ich zu meiner Cousine auf Mallorca. Unter anderem bin ich dort zu ihrem Geburtstag eingeladen. Ich habe genug entsprechende Abendkleider, aber gestern habe ich einen exklusiven dunkelgrünen Rock gekauft. Der ist so schön und schick! Aber dazu passt keine einzige Bluse aus meinem Kleiderschrank.

- Beschreiben Sie mir bitte Ihren neuen Rock.

- Das ist ein extravaganter asymmetrischer Rock aus gecrashter reiner Seide, der bestimmt auffällig ist. Der Rock ist allover bedruckt und hat mehrere dreieckige Einsätze im Saum. Der schwarz-oliv-goldfarbene Druck ist sehr interessant und apart. Insgesamt sieht dieser Rock herrlich leicht und luftig aus.

- Das alles klingt sehr interessant, für uns sind aber in erster Linie die Farben und das Material wichtig. Da Sie einen Rock aus Seide haben, werde ich Ihnen zuerst alle unsere Seidenblusen anbieten.

- Sie haben Recht, ich möchte unbedingt eine Bluse aus Seide finden.

- Also, unsere Farben sind Schwarz, Oliv und Gold.

Da Ihr Rock allover bedruckt ist, sehen wir uns gleichfarbige und zweifarbige Blusen an.

- Machen wir! Ich bin gespannt!

- Dieses herrliche goldfarbene Carmenshirt mit elastischem Bund kann über die Schultern getragen werden. Für Mallorka wäre das eine sehr passende Variante. Edle Paillettenstickerei macht die Bluse richtig glamourös und auffallend.

- Diese Bluse ist sehr schön und gefällt mir sehr gut. Ich habe bis jetzt keine Carmenshirts gehabt.

- Probieren Sie diese Bluse unbedingt an!

- Sie hat einen super Schnitt, der nicht alltäglich aussieht, und sitzt schön locker. Und das Material ist unwahrscheinlich zart.

- Das ist doch reine Seide!

- Ich verstehe. Ich kaufe diese Bluse eindeutig.

- OK. Hier ist die zweite Bluse, die unglaublich raffiniert ist. Sie ist auch aus reiner Seide und feiner Spitze gemacht. Modische feminine asymmetrische Fledermausärmel und breiter Spitzenabschluss machen sie echt schick. Probieren sie diese schwarze Bluse mit goldfarbigen Kontrastnähten und Spitze!

- Die Bluse ist traumhaft. Sie ist einfach genial! Also, wie sie richtig festgestellt haben, habe ich Größe 40. Die Bluse fällt aber etwas kleiner aus. Könnten Sie mir bitte Größe 42 geben?

- Glücklicherweise haben wir auch Größe 42. Sie sitzt einwandfrei!

- Herzlichen Dank! Jetzt habe ich alles für meine Reise. Beide Blusen sind traumhaft schön und passen zu meinem neuen Rock.

- Nicht zu danken. Ich schicke die Ware an die Kasse drüben. Nach der Bezahlung bekommen Sie ihre Ware.

- Vielen Dank! Auf Wiedersehen! Schönen Tag noch!

- Auf Wiedersehen! Gleichfalls!

*Игра 2. Студенты получают карточки, на каждой начало предложения. Задача – продолжить предложение, используя инфинитивную конструкцию.*

| | | |
|---|---|---|
| 1. | Ich bin glücklich,... | что вижу тебя в нашем доме вновь; что познакомился с таким замечательным человеком. |
| 2. | Der Vater bedauert sehr,... | что не может проводить (bringen) нас на вокзал; что незаслуженно (unverdient) наказал сына. |
| 3. | Man hofft immer,... | в последнюю минуту найти верное решение; что был прав. |
| 4. | Kolumbus glaubte,... | что может добраться до Индии морским путем (auf dem Seeweg); что нашел морской путь в Индию. |
| 5. | Die Touristen fürchteten,... | что могут опоздать на поезд; что приехали слишком поздно. |
| 6. | Sie ist überzeugt, | что помогает своим друзьям; что поступила верно. |
| | | |

*Игра 3. Студенты получают карточки, на них два предложения. Задача – продолжить предложение, используя инфинитивную конструкцию.*

1. Wir bleiben im Wartesaal. Wir wollen auf den Zug warten.

2. Die Touristen gingen ins Restaurant. Sie wollten zu Mittag essen.

3. Du fährst aufs Land. Du willst dich nach dem schwierigen Semester erholen?

4. Meine Freundin kam zu mir. Sie wollte mir zum Geburtstag gratulieren.

5. Ihr Sohn fährt ins Ausland. Er will dort studieren.

6. Ich lese alle Zeitungsanzeigen. Ich will eine billigere Wohnung mieten.

7. Die Mutter backte einen Obstkuchen. Sie wollte die Gäste ihrer Tochter damit bewirten.

8. Wir treffen uns heute um 18 Uhr. Wir wollen alle Streitfragen erörtern.

9. Du musst dich beeilen. Du willst den Zug erreichen.

10. Man muss viel lesen. Man will seine Kenntnisse erweitern.

1. Ich schicke dir Heber ein Telegramm. Ich soll dich anrufen.

2. Wir blieben noch eine Woche in St.Petersburg. Wir sollten am Montag zurückfahren.

3. Der Kranke ging zur Arbeit. Er sollte sich an den Arzt wenden.

4. Du ziehst dich so langsam an. Du sollst dich beeilen.

5. Der Junge steckte seine Mütze in die Tasche. Er sollte sie aufsetzen.

6. Er ist Advokat geworden. Er sollte das Familiengeschäft weiterführen.

7. Meine Tochter telefoniert den ganzen Abend mit ihren Freunden. Sie soll sich nach der Abflugzeit erkundigen.

8. Warum schickst du deinen Eltern nur eine Postkarte? Du solltest ihnen doch einmal einen richtigen Brief schreiben.

9. Sie sind wahrscheinlich schon im Zuschauerraum. Sie sollten auf uns beim Eingang warten.

10. Warum fahrt ihr immer mit dem Bus? Ihr sollt beim schönen Wetter ein Stück zu Fuß laufen.

11. Die Schülerin sieht die Lehrerin an. Sie beantwortet ihre Frage nicht.

12. Die alte Frau saß am Fenster. Sie nahm am Gespräch nicht teil.

13. Der Sohn schwieg. Er widersprach ihr nicht.

14. Das Mädchen gibt das Buch in die Bibliothek zurück. Sie hat es nicht gelesen.

15. Warum verlässt du das Zimmer? Du hast dich von unseren Freunden nicht verabschiedet.

16. Mein Bruder trifft seine Entscheidung. Er hat es nicht lange überlegt.

17. Man darf sich nicht ans Steuer setzen. Man besitzt keinen Führerschein.

18. Der Ladeninhaber kündigt einem der Verkäufer. Er hat ihn davon rechtzeitig nicht informiert.

19. Die Studenten dürfen in Deutschland nicht arbeiten. Sie haben keine Arbeitserlaubnis.

20. Das Mädchen verließ das Elternhaus. Sie hat ihren Eltern kein einziges Wort davon gesagt.

21. Er steckte das Geld in seine Geldtasche. Er zählte es nicht nach.

22. Meine Großmutter heiratete einst meinen Großvater. Sie hat die Einwilligung ihrer Eltern nicht bekommen.

## Практическое занятие №6. РОЛЕВАЯ ИГРА «НА ПРИЕМЕ У ВРАЧА», «СЛОЖНОЕ ПРЕДЛОЖЕНИЕ»

***Цель работы***: *Усвоение речевых формул, лексики по теме: «медицина», «здоровье»; формирование умений построения монологического текста-повествования, а также обиходно-бытового диалога.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание: «Посетить дантиста», «Посетить терапевта», «Выписать лекарство от кашля» и под. Далее каждая группа представляет свой диалог, они пользуются и приведенными ниже диалогами и текстами.*

Guten Tag, Herr Schulz.

- Guten Tag. Setzen Sie sich bitte. Haben Sie irgendwelche konkrete Beschwerden bzw. Schmerzen?

- Gestern habe ich mich wohl gefühlt. Am Abend sind ich und mein Kollege etwas länger im Büro geblieben. Als ich nach Hause kam, habe ich festgestellt, dass ich unheimlich müde bin. Ich konnte mich sogar nicht duschen und hatte

keinen Appetit. Ich ging sofort zu Bett. Heute bin ich ganz müde aufgestanden. Ich habe kein Fieber, aber der Kopfschmerzen ist unerträglich. Also, ich fühle mich eher krank, als gesund.

- Sie sehen auch nicht gesund aus. Man merkt sofort, dass Sie sich schlecht fühlen. Ziehen Sie sich bitte bis auf den Schlüpfer aus. Ich muss Sie gründlich untersuchen.

- Ich habe vergessen, zu sagen, dass mein Magen mir auch etwas weh tut.

- Haben Sie irgendwelche Probleme mit dem Herzen?

- Bisher hat man bei mir keine festgestellt.

- Jetzt messen wir Ihren Blutdruck. Also, Ihr Blutdruck ist richtig erhöht. Dies erklärt das Aufkommen von Ihren Kopfschmerzen.

- Werde ich krankgeschrieben? Ich muss in drei Tagen auf eine wichtige Dienstreise fahren.

- Dienstreisen sind innerhalb von drei-vier Wochen völlig ausgeschlossen. Sie bekommen von mir eine Einweisung zu unserem Kardiologen. Er macht Ihr Elektrokardiogramm und stellt eine genaue Diagnose. Mit den Resultaten kommen Sie danach wieder hierher und wir sprechen über die weitere Behandlung.

- Was soll ich inzwischen tun?

- Inzwischen müssen Sie möglichst viel schlafen, sich ausruhen und mindestens drei Stunden pro Tag im Freien verbringen.

- Muss ich irgendwelche Medikamente einnehmen?

- Jetzt verschreibe ich Ihnen ein paar Beruhigungsmittel und nach der vollständigen Untersuchung bekommen Sie die ganze Liste mit unseren Hinweisen und Empfehlungen.

- Das alles klingt etwas traurig und ändert alle meine Pläne für die nächste Zukunft.

- Das stimmt, aber jetzt müssen Sie auf Ihre Gesundheit gut aufpassen. Ihre heutigen Bemühungen werden Ihnen ermöglichen, viel größere Probleme in der Zukunft zu vermeiden.

- Vielen Dank, ich habe alles verstanden. Ich werde alle Ihre Vorschriften erfüllen und hoffe, dass meine Krankheit bald vorbei ist.

- Ich meinerseits wünsche Ihnen gute Besserung und bin sicher, dass Sie bald über Ihre Kopfschmerzen und andere unangenehme Gefühle vergessen. Hier ist Ihr Krankenschein. Auf Wiedersehen!

- Auf Wiedersehen, Herr Schulz!

***

Telefonanruf:

- Hallo! Ich brauche dringend einen Zahnarzt. Mein Zahn tut mir so weh, dass ich nichts essen und trinken kann. Kann ich schnellst möglich einen Termin machen?

- Selbstverständlich. Ihre Situation ist außerordentlich. Sie müssen in unsere Klinik kommen und ein kleines Formular ausfüllen. Dann begleitet die Krankenschwester Sie sofort und direkt zu Ihrem Zahnarzt.

***

Im Behandlungsraum:

- Schönen guten Morgen!

- Guten Morgen! Nehmen Sie bitte Platz und erzählen Sie, welche Beschwerden Sie haben.

- Gestern hat meine Frau Süßkirschen gekauft. Ich las das Buch und aß diese verdammten Beeren. Ich war irgendwo ganz tief in meinen Gedanken und habe meine Zähne unwillkürlich fest zusammengepresst. Ein zufälliger Kirschkorn blieb aber zwischen den Zähnen. Dieser Kirschkorn hat mir zwei Zähne auf einmal gebrochen.

- Machen Sie bitte Ihren Mund auf. Der obere Zahn war völlig gesund. Von diesem Zahn hat der Korn ein nicht besonders großes Stück abgeschlagen. Das ist alles. Ich mache eine entsprechende Füllung und das Problem ist gelöst.

Unten sieht alles anders aus.

- Ausgerechnet unten habe ich furchtbare Schmerzen. Und ich habe im Spiegel gesehen, dass von meinem unteren Zahn kaum was geblieben ist. Werden Sie diesen Zahn rausziehen?

- Der untere Zahn ist stark zerstört, aber die Zahnerhaltung ist möglich. Ich muss die entzündeten Kanäle behandeln, auffüllen und danach den Zahn verschließen. Dafür werde ich spezielle Aufbaumaterialien verwenden.

- Sie haben so viele Schritte aufgelistet, die gemacht werden sollen. Bestimmt werde ich Sie nicht nur einmal besuchen müssen.

- Ja, das wird zwei-drei Wochen dauern. Die Kanäle müssen grundsätzlich gereinigt werden, um künftige Entzündungen auszuschließen. Den richtig behandelten Zahn werden wir dann mit einer Krone verschließen, weil so eine große Füllung keine sichere Versiegelung gewährleistet.

- Aus welchem Material wir meine Krone sein? Welche Farbe wird Sie haben? Oh, Gott! Nur kein Metall, bitte!

- Beruhigen Sie sich bitte! Ihre neue Krone wird ganz natürlich aussehen und dadurch Ihren Wünschen und Vorstellungen über schöne Zähne optimal entsprechen. Ich mache eine Vollkeramikkrone aus einem erstklassigen Material. Sie werden keine Abweichungen in Farbe und Form merken, wenn Sie die Krone mit Ihren eigenen Zähnen vergleichen.

- Danke, Doktor. Starten wir die Behandlung. Ich bin mit allen Ihren Vorschlägen völlig einverstanden.

***

- Guten Morgen, Herr Doktor!

- Guten Morgen! Kann Ich Ihr helfen? Wo tut's denn weh?

- Ich brauche einen Internist. Meiner Meinung nach, habe Ich mich erkältet. Ich habe starke Halsschmerzen. Mir ist sehr schwindlig.

- Husten Sie?

- Ja. Seit gestern.

- Haben Sie Fieber?

- Ich weiß nicht. Aber mir ist erbärmlich zumute.

- Machen Sie den Oberkörper frei. Ich muss Sie untersuchen. Atmen Sie tief durch. (In eine Minute) Jetzt halten Sie den Atem an… Ich bin sicher, dass Ihre Lungen gesund sind. Bitte, machen Sie den Mund auf. Das hab ich mir gedacht! Sie haben Grippe.

- Grippe?! Wirklich?

- Alles ist nicht so schlecht. Sie müssen nur im Bett bleiben und meinen Empfehlungen folgen. Die Besserung ist eine Frage der Zeit.

- Welche Arznei empfehlen Sie? Wann soll ich sie einnehmen? Vor dem Essen? Nach dem Essen?

- Hier ist das Rezept. Alles notwendige ist drin genannt. Gehen Sie damit zur Apotheke, um diese Tabletten zu kaufen. Sie sind nicht teuer.

- Muss ich Sie noch mal besuchen?

- Zweifelllos, wenn sie ein Krankenschein bekommen wollen. Studieren oder arbeiten Sie?

- Ich bin Student. Hier ist meine Karte.

- Gut. Kommen Sie in acht Tagen. Ich wünsche Ihnen gute Besserung.

- Danke.

- Alles Gute. Auf Wiedersehen!

- Auf Wiedersehen!

*Игра 2. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить сообщение о сложносочиненном предложении с определенными союзами. Затем каждая группа представляет свое сообщение, сопровождая его максимальным количеством примеров.*

**Карточка 1**

1) ***und*** «и»: *Hier gibt es Zeitungen und Zeitschriften.- Здесь есть газеты и журналы*.

2) ***auch*** «также, тоже, и; даже»: *Ich kenne ihn auch.- Я его тоже знаю. So hat er auch gemacht.- Так он и сделал. Auch der kleinste Fehler darf nicht übersehen werden.- Даже малейшая ошибка не может быть пропущена*.

3) ***sowie*** «(равно) как и, а также»: *Hier gibt es Zeitungen und Zeitschriften sowie Broschüren und Bücher.- Здесь есть газеты и журналы, а также брошюры и книги*.

4) ***sowohl ... als auch (sowohl ... wie auch)*** «и... и», «как ... так и»: *sowohl mein Bruder als auch ich - и мой брат, и я; как мой брат, так и я; Er kannte sowohl die Stadt selbst als auch (wie auch) ihre Umgebung genau.- Он хорошо знал, как сам город, так и его окрестности*.

5) ***nicht nur ... sondern auch*** «не только, но и»: *Er lebt hier nicht mir im Sommer, sondern auch im Winter.- Он живет здесь не только летом, но и зимой*.

6) ***außerdem*** «кроме того»: *Er ist klug und außerdem sehr fleißig.- Он умен и кроме того очень прилежен*.

7) ***und zwar*** «а именно; и притом»: *Kommen Sie morgen, und zwar um 2 Uhr.- Приходите завтра, а именно в 2 часа. Er macht das, und zwar sofort.- Он сделает это и притом немедленно*.

8) ***weder ... noch*** «ни ... ни»: *Weder er noch ich können morgen kommen.- Ни он, ни я не можем завтра прийти*.

9) ***darum, deshalb, deswegen*** «поэтому»: *Ich habe viel zu tun, darum kann ich nicht mit Ihnen gehen.- У меня много дел, поэтому я не могу пойти с вами*.

**Карточка 2**

1) ***bald ... bald*** «то ... то»; ***bald so, bald anders*** - то так, то иначе: *Bald regnete es, bald schneite es.- То шел дождь, то снег*.

2) ***dann*** «затем»: *Zuerst lese ich den Text, dann übersetze ich ihn.- Сначала я читаю текст, затем я перевожу его*.

3) ***aber*** «но, однако»: *Dieses Kapitel ist kurz, aber wichtig.- Эта глава короткая, но (однако) важная*.

4) **allein** «но, однако»: *Er musste bald kommen, allein wir konnten nicht länger warten.- Он должен был скоро прийти, однако мы не могли дольше ждать*. Обратите внимание на многозначность allein. Помимо значения «но», allein имеет значение «один, одна, одно, одни» и значение «только»; *Er war gestern allein zu Hause.- Он вчера был один дома, Allein er kann uns helfen. - Только он может помочь нам-*

5) ***und*** «а»: *Alle gehen, und ich soll bleiben.- Все уходят, а я должен остаться*.

6) ***sondern*** «а»: *Er ist nicht Student, sondern Aspirant.- Он не студент, а аспирант***.**

7) ***sonst*** «а то, иначе»: *Beeilen Sie sich, sonst kommen Sie zu spät.- Поторопитесь, а то (иначе) вы опоздаете*.

8) ***doch*** «однако, но; все-таки, все же»: *Er wollte kommen, doch sein Vater wurde krank.- Он хотел прийти, но заболел его отец. Er hat es versprochen, aber hat es doch nicht gemacht.- Он обещал, но все же не сделал этого*.

9) ***daher*** «и (а) поэтому, а потому»: *Die Kritik ist gerecht, daher sollen wir anders arbeiten.- Критика справедлива, и поэтому мы должны работать иначе*.

**Карточка 3**

1) ***jedoch, dennoch*** «однако, все-таки, тем не менее»: *Er ist sehr beschäftigt, jedoch hilft er mir.- Он очень занят, тем не менее (однако, все-таки) он помогает мне*.

2) ***trotzdem*** «несмотря на это, все же»: *Es regnete, trotzdem kam er zu uns.- Шел дождь, несмотря на это (все же) он пришел к нам*.

3) ***zwar*** «правда, хотя (и)»: *Er kam zwar, doch war es zu spät.- Он хотя и пришел, но было слишком поздно. Он, правда, пришел, но было слишком поздно*.

4) ***oder*** «или»: *Wir fahren heute oder morgen.- Мы поедем сегодня или завтра. Wählen Sie das eine oder das andere.- Выберите то или другое*.

5) ***entweder ... oder*** «или ... или, либо ... либо»: *Entweder kommt er, oder er ruft an.- Он или придет, или позвонит по телефону. Он либо придет, либо позвонит по телефону*.

6) ***denn*** «так как, потому что, ибо»: *Er spricht gut deutsch, denn er lebt schon lange in Leipzig.- Он хорошо говорит по-немецки, так как уже давно живет в Лейпциге*.

7) **nämlich** «дело в том, что; так как, ведь»: *Ich konnte ihn nicht sehen, er ist nämlich verreist.- Я не смог его увидеть, дело в том, что он уехал*.

8) ***also*** «итак, так; следовательно, стало быть, значит»: *Hier sind seine Sachen, also ist er hier gewesen.- Вот его вещи, следовательно, он был здесь*.

9) **folglich** «следовательно, поэтому, итак»: *Die Sachen sind nicht gebracht worden, folglich müssen wir sie holen.- Вещи не принесли, следовательно (поэтому, итак) мы должны их доставить сами*.

## Практическое занятие №7. РОЛЕВАЯ ИГРА «В Офисе», «АНАЛИЗ ПРИРОДНЫХ ЗОН ГЕРМАНИИ»

***Цель работы***: *Усвоение лексики по теме: «деловое общение», «окружающая среда», «природные зоны»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить диалог двух коллег в офисе. При подготовке студенты используют приведенные ниже диалоги*

- Anette: Guten Morgen, Doris! Es ist noch so früh und du bist schon an deinem Arbeitstisch!
- Doris: Üblicherweise fahre ich mit der U-Bahn und heute hat mich meine Nachbarin mit ihrem Auto ins Büro gebracht. Ich bin schon eine halbe Stunde da.
- Anette: Alles klar. Hast du unsere eingegangenen E-Mails durchgesehen?
- Doris: Es gibt keine interessanten Angebote und keine wichtigen Briefe.
- Anette: OK. Ich stelle jetzt eine Liste unserer Kunden zusammen, an die wir unseren neuen Dienstleistungsvertrag schicken müssen.
- Doris: Unser Chef hat mich grade angerufen und mitgeteilt, dass er erst am Nachmittag ins Büro kommt. Er hat einen wichtigen Termin mit Deutscher Bank vereinbart. Er möchte die Möglichkeiten der Kreditierung unserer Kunden besprechen.
- Anette: Wenn Deutsche Bank uns entgegen kommt, wird das die weitere Erhöhung unserer Geschäftsvolumen fördern.
- Doris: Stimmt, unsere Firma könnte dann ihre Positionen auf dem Markt stärken.
- Anette: Das Telefon klingelt! Könntest du bitte den Anruf beantworten?
- Doris: Das ist bestimmt eine Faxnachricht von unserem Partner. Hörst du, das Telefon klingelt nicht mehr und die Faxnachricht wird automatisch empfangen.
- Anette: Du hast Recht. Und für dich habe ich für heute eine wichtige Aufgabe. Du musst die im Büro vorhandenen Kanzleiwaren überprüfen und danach alles Notwendige für unsere weitere ununterbrochene Tätigkeit bestellen.
- Doris: OK, mache ich gleich. Also, wir haben praktisch keine Druckerpapiervorräte, keine Heftklammer, wenig Büroordner und Aktenmappen. Wie immer müssen schwarze und blaue Kugelschreiber und Bleistifte bestellt werden. Unser Chef hat gebeten, eine Notebook-Unterlage mit Kühler für ihn zu bestellen. Meine Tischlampe ist kaputt, also ich brauche eine neue. Dann müssen wir Toilettenpapier, Handwaschmittel und Kaffeefilter nicht vergessen.
- Anette: Bitte, bestelle ein neues USB-Kabel für mein Handy und überschreibbare CD- und DVD-Discs.
- Doris: OK. Ich bin mit der Auflistung fertig. Jetzt rufe ich unseren Lieferanten von Kanzleiwaren und bestelle alles.
- Anette: Ja, bitte. Und danach können wir in unsere Kantine gehen.
- Doris: Gerne!

*Игра 2. Студенты делятся на небольшие группы по 2-3 человека. Каждая группа получает задание подготовить сообщение об одной из природных зон Германии. Далее каждая группа, используя Интернет, тексты, приведенные ниже и материалы учебника готовит сообщение. Затем каждая группа представляет свою зону и отвечает на вопросы студентов и преподавателя.*

*Wetter und Klima*

- Rita: Hallo, Loretta! Was machst du in der Stadt bei so einem schönen Wetter? Wir haben doch Ferien!
- Loretta: Meine Schwester Helga ist krank. Sie hat Feuchtblattern mit einem Riesenausschlag und Fieber. Leider fahren wir aufs Land erst in ein paar Wochen, wenn alles vorbei ist.

- Rita: Aber ich habe im Wetterbericht gehört, dass die zweite Julihälfte nicht so schön und warm, wie die erste sein wird. In den ersten zehn Tagen wird es heiter bis wolkig sein, dabei werden die Tagestemperaturen durchschnittliche Monatsnormen übersteigen.
- Loretta: Und was sagt man über die zweite Julihälfte?
- Rita: Nach der ersten Dekade werden die Tagestemperaturen wesentlich sinken, die Nächte werden ziemlich kühl und es wird ständig regnen. Ich habe verstanden, dass wir einen für unsere Region untypisch kalten und regnerischen Juli haben werden.
- Loretta: Das klingt nicht so gut. Hoffentlich irren sich unsere Synoptiker und wir werden noch schöne Sommertage erleben.
- Rita: Man sagt nicht, dass der Sommer schon zu Ende ist. Man sagt nur, dass wir alle auf warme und sonnige Tage ziemlich lange warten müssen.
- Loretta: In unserer Region ist das Klima ziemlich mild. Aber das bedeutet nicht, dass unser Sommer lange dauert. Wenn wir im Juli kein gutes Wetter haben, so wird es bedeuten, dass der Sommer in diesem Jahr viel kürzer sein wird.
- Rita: Du hast Recht. Aber wir haben immer noch Altweibersommer im September. Das ist natürlich kein richtiger Sommer, aber diese Tage sind auch sehr warm, sonnig und angenehm.
- Loretta: Ja, und ich hoffe, dass wir noch viele schöne Tage in diesem Jahr haben. Meine Schwester wird bald gesund und wir werden noch viel Zeit in unserem Landhaus verbringen.
- Rita: Ich wünsche euch alles Gute und deiner Schwester gute Besserung!
- Loretta: Vielen Dank! Und bis später!
- Rita: Tschüss!

***

Bodensee

Unter der Bezeichnung Bodensee fasst man die Gewässer Obersee (eigentlicher Bodensee) und Untersee zusammen. Sie liegen im nördlichen Alpenvorland, werden vom Rhein durchflossen und verteilen sich auf Deutschland, die Schweiz und Österreich. Es handelt sich also um zwei Seen und einen sie verbindenden, nur 4 Kilometer langen Fluss, den Seerhein in Konstanz mit wiederum eigener seeartiger Verbreiterung. Der bis vor wenigen Tausend Jahren noch zusammenhängende See wurde durch die Tiefenerosion des Hochrheins, die den Seespiegel absinken und die Konstanzer Schwelle hervortreten ließ, getrennt. Die beiden Seen trugen in der Antike noch unterschiedliche Namen, danach entwickelte sich aus unbekannten Gründen der gemeinsame Name. Der Artikel behandelt nicht nur die Gewässer an sich, sondern auch die umgebende Bodenseeregion, die sich nicht immer mit dem Bodenseebecken deckt.

Der Bodensee liegt im Alpenvorland. Die Uferlänge beider Seen beträgt 273 km. Davon liegen 173 km in Deutschland (Baden-Württemberg 155 km, Bayern 18 km), 28 km in Österreich und 72 km in der Schweiz. Der Bodensee ist, wenn man Obersee und Untersee zusammenrechnet, mit 536 km² nach dem Plattensee (594 km²) und dem Genfersee (580 km²) flächenmäßig der drittgrößte, gemessen am Wasservolumen (48,5 km³) nach dem Genfersee (89 km³) der zweitgrößte See Mitteleuropas und erstreckt sich zwischen Bregenz und Stein am Rhein über 69,2 km. Sein Einzugsgebiet beträgt rund 11.500 km² und reicht im Süden bis nach Italien.

Die Fläche des Obersees beträgt 473 km². Er erstreckt sich zwischen Bregenz und Bodman-Ludwigshafen über 63,3 km und ist zwischen Friedrichshafen und Romanshorn 14 km breit; an seiner tiefsten Stelle zwischen Fischbach und Uttwil misst er 254 m.

Die drei kleinen Buchten des Vorarlberger Ufers haben Eigennamen: Vor Bregenz liegt die Bregenzer Bucht, vor Hard und Fußach die Fußacher Bucht und westlich davon der Wetterwinkel. Weiter westlich, bereits in der Schweiz, befindet sich die Rorschacher Bucht. Nördlich, auf bayerischer Seite, ist die Reutiner Bucht. Der Bahndamm vom Festland zur Insel Lindau und die Seebrücke für den Autoverkehr grenzen vom Bodensee den so genannten „Kleinen See“ ab, der zwischen dem Lindauer Ortsteil Aeschach und der Insel liegt.

Der nordwestliche, fingerförmige Arm des Obersees heißt Überlinger See. Im allgemeinen Sprachgebrauch wird der Überlinger See als eigenständiger Seeteil betrachtet, die Grenze zwischen Obersee und Überlinger See verläuft in etwa entlang der Linie zwischen der Südostspitze des Bodanrücks (das zur Stadt Konstanz gehörende „Hörnle“) und Meersburg. Östlich vor Konstanz liegt der sogenannte Konstanzer Trichter zwischen dem deutschen und dem Schweizer Ufer.

Der Untersee, der vom Obersee bzw. von dessen nordwestlichem Arm Überlinger See durch die große Halbinsel Bodanrück abgetrennt ist, weist eine Fläche von 63 km² auf. Er ist durch die Endmoränen verschiedener Gletscherzungen und Mittelmoränen geprägt und stark gegliedert. Diese Seeteile haben eigene Namen. Nördlich der Insel Reichenau befindet sich der Gnadensee. Westlich der Insel Reichenau, zwischen der Halbinsel Höri und der Halbinsel Mettnau befindet sich der Zeller See. Nördlich der Mettnau liegt der Markelfinger Winkel. Die Drumlins des südlichen Bodanrücks setzen sich am Grund dieser nördlichen Seeteile fort. Südlich der Reichenau erstreckt sich von Gottlieben bis Eschenz der Rheinsee mit seiner zum Teil ausgeprägten Rheinströmung. Früher wurde dieser Seeteil nach dem Ort Berlingen Bernanger See genannt. Auf den meisten Karten ist der Name des Rheinsees auch deshalb nicht aufgeführt, weil sich dieser Platz am besten für die Beschriftung des Untersees eignet.

Das Bodenseebecken wurde wesentlich während der Würm-Eiszeit durch den aus dem alpinen Rheintal austretenden Rheingletscher geformt, in dessen fluvioglazial erodiertem Zungenbecken der heutige Bodensee liegt. Dieser kann insofern als würmglazialer Zungenbeckensee oder Gletscherrandsee bezeichnet werden. Nach der Eiszeit bestand der Bodensee zuerst als ein See. Der Seerhein und die damit verbundene Trennung in zwei Seen entstand vor mehreren tausend Jahren durch die rheinische Erosion, die den Seespiegel absenkte und das heutige Seerheintal trockenlegte.

Wie jeder glaziale See wird auch der Bodensee durch Sedimentation in geologisch naher Zukunft verlanden. Dieser Prozess lässt sich am besten an den Mündungen größerer Flüsse, vor allem der des Alpenrheins, beobachten. Die Verlandung wird beschleunigt durch die stets weitergehende rheinische Erosion und die damit verbundene Absenkung des Seespiegels.

Hauptzufluss des Obersees ist der Alpenrhein, Abfluss des Obersees ist der Seerhein, der wiederum Hauptzufluss des Untersees ist. Abfluss des Untersees ist der Hochrhein. Der Alpenrhein und der Seerhein vermischen sich nur bedingt mit den Seewässern und durchströmen die Seen in meist gleich bleibenden Bahnen. Daneben gibt es zahlreiche kleinere Zuflüsse (236). Die wichtigsten Nebenzuflüsse des Obersees sind Bregenzer Ach, Leiblach, Argen, Schussen, Rotach, Seefelder Aach, Stockacher Aach, Aach (bei Arbon), Steinach, Goldach, Dornbirner Ach und Alter Rhein. Wichtigster Nebenzufluss des Untersees ist die Radolfzeller Aach.

Im Bodensee liegen zehn Inseln größer als 2000 m². Die größte Insel ist die Reichenau. Die größten im Obersee sind die Mainau und die Insel Lindau.

Auf der Insel Reichenau, die zur Gemeinde Reichenau gehört, liegt das ehemalige Kloster Reichenau. Dieses gehört auch aufgrund dreier früh- und hochmittelalterlicher Kirchen zum Welterbe der UNESCO. Die Insel ist auch durch intensiv betriebenen Gemüsebau bekannt.

Die Insel Mainau liegt im Südosten des Überlinger Sees. Die Eigentümer, die Familie Bernadotte, haben die Insel als touristisches Ausflugsziel eingerichtet und dafür botanische Anlagen und Tiergehege geschaffen.

Auf der Insel Lindau ganz im Osten des Obersees befindet sich sowohl die Altstadt als auch der Hauptbahnhof der gleichnamigen Stadt Lindau.

Kleinere Inseln im Obersee sind die Dominikanerinsel (durch einen sechs Meter breiten Graben von der Altstadt von Konstanz getrennt) mit dem Steigenberger-Hotel (2 ha) und die winzige Insel Hoy bei Lindau, im Untersee die Insel Werd im Übergang zum Hochrhein, zwei kleine Inseln vor dem Wollmatinger Ried (Triboldingerbohl mit 13 ha und Mittler oder Langbohl mit 3 ha) und die so genannte Liebesinsel (0,2 ha) südwestlich der Halbinsel Mettnau. Alle genannten Inseln im Überblick, von Ost nach West:

Insel Werd, Mittleres Werdli und Unteres Werdli bilden die Gruppe der Werd-Inseln und liegen am Ausfluss des Rheins aus dem Untersee bei Stein am Rhein in den Hochrhein. Sie sind die einzigen Bodenseeinseln, die zur Schweiz gehören.

In den Bodensee ragen einige Halbinseln unterschiedlicher Größe.

- Der Bodanrück, die größte Halbinsel, trennt den Obersee (Seeteil Überlinger See) vom Untersee. Er erstreckt sich über eine Fläche von 112 km².
- Die Mettnau im Untersee, die sich der Insel Reichenau entgegenstreckt, trennt den Zeller See im Süden vom Markelfinger Winkel im Norden. Sie hat eine Flächenausdehnung von 1,7 km².
- Die etwa 45 km² große Höri, die sich ebenfalls der Insel Reichenau entgegenstreckt, trennt den Zeller See im Norden vom Rheinsee im Süden.
- Im Südosten, nahe der Mündung des neuen Rheinkanals, ragt der Rohrspitz mit einer Fläche von etwa 50 ha rund 1,2 km in den See und bildet die westliche Umrandung der Fußacher Bucht.

- Die Halbinsel Wasserburg mit dem Schloss Wasserburg und der Pfarrkirche St. Georg im nordöstlichen Obersee liegt zwischen der Nonnenhorner Bucht im Westen und der Wasserburger Bucht im Osten. Sie hat eine Flächenausdehnung von 2,3 ha und war eine Insel bis 1720, als die Fugger einen Damm aufschütteten. Im März 2009 lebten 27 Einwohner auf der Halbinsel.
- Die Galgeninsel in der Reutiner Bucht ist ebenfalls eine Halbinsel, die früher eine Insel war. Sie ist nur 0,16 ha groß.

Das Ufer des Bodensees besteht überwiegend aus Kies. An einigen Stellen findet man aber auch echten Sandstrand, so am Rohrspitz im österreichischen Abschnitt des Sees oder bei der Marienschlucht.

Das Bodenseeklima ist durch milde Temperaturen mit gemäßigten Verläufen (durch die ausgleichende und verzögernde Wirkung des Wasservolumens) gekennzeichnet. Es gilt allerdings – aufgrund des ganzjährigen Föhneinflusses, häufigen Nebels im Winterhalbjahr und auftretender Schwüle im Sommer – als Belastungsklima.

Der Bodensee gilt bei Wassersportlern aufgrund der Gefahr starker Sturmböen bei plötzlichen Wetterwechseln als nicht ungefährliches und anspruchsvolles Binnenrevier. Gefährlichster Wind ist der Föhn, ein warmer Fallwind aus den Alpen, der sich insbesondere durch das Rheintal auf das Wasser ausbreitet und bei teils orkanartigen Windstärken typische Wellenberge mit mehreren Metern Höhe vor sich hertreiben kann.

Ähnlich gefährlich sind die für Ortsunkundige u. U. völlig überraschend auftretenden Sturmböen bei Sommergewittern. Sie fordern immer wieder Opfer unter den Wassersportlern. Bei einem Sturm im Juli 2006 während eines Gewitters wurde eine Wellenhöhe von bis zu 3,50 Metern erreicht.

Aus diesen Gründen gibt es ein über alle drei Anrainerländer verknüpftes Sturmwarnsystem: Der Bodensee ist für Sturmwarnungen in drei Warnregionen (West, Mitte, Ost) aufgeteilt. Für jede Region kann eine Starkwind- oder Sturmwarnung ausgegeben werden. Eine Starkwindwarnung erfolgt bei erwarteten Windböen zwischen 25 und 33 Knoten beziehungsweise 6 bis 8 Windstärken nach der Beaufortskala. Eine Sturmwarnung kündigt die Gefahr von Sturmwinden mit Geschwindigkeiten ab 34 Knoten beziehungsweise 8 Windstärken nach der Beaufortskala an. Um diese Warnungen bekannt zu machen, sind rund um den See orangefarbige Blinkscheinwerfer installiert, die bei Starkwindwarnung mit einer Frequenz von 40 Mal pro Minute, bei Sturmwarnung 90 Mal pro Minute blinken. Dabei kann es wegen unterschiedlich geregelter Zuständigkeiten und Einschätzungen durchaus vorkommen, dass am Schweizer Ufer des Obersees schon Sturmwarnung einsetzt, am deutschen oder österreichischen Ufer aber noch nicht (und umgekehrt). Die Bodenseeschiffe und die Fähren signalisieren eine Sturmwarnung durch einen am Masten hochgezogenen Sturmballon.

Ein Jahrhundertereignis ist die Seegfrörne des Bodensees, wenn Untersee, Überlinger See und Obersee komplett zugefroren sind, so dass man den See überall sicher zu Fuß überqueren kann. Die drei letzten so genannten Seegfrörne waren im Jahr 1963, 1880, 1830.

Bestimmte Teile des Untersees frieren hauptsächlich aufgrund der geringen Wassertiefe und der geschützten Lage häufiger zu, wie z. B. der sogenannte Markelfinger Winkel zwischen der Gemeinde Markelfingen und der Halbinsel Mettnau.

## Практическое занятие № 8. КОНКУРС ПРОЕКТОВ «Виды предприятий в германии», «ГОРОДА ГЕРМАНИИ»

***Цель работы***: *Усвоение лексики по теме: «город», «природные зоны»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Проект. Все студенты заранее получили задание подготовить сообщение об одном из предприятий или городов Германии, включающем письменный текст, устный доклад и инфографику. На занятии каждый представляет свой проект и отвечает на вопросы студентов и преподавателя.*

### Einzelunternehmung

Eine Einzelunternehmung hat, wie der Name es bereits verrät, nur einen alleinigen Inhaber und stellt in Deutschland die meistgenutzte Rechtsform dar. Diese Rechtsform ist besonders gut geeignet für kleine und mittlere Unternehmungen. Ist der Einzelunternehmer ein Kaufmann, muss die Firma den Zusatz "eingetragener Kaufmann" bzw. "eingetragene Kauffrau" (e.K.) tragen.

Der Alleininhaber hat alle Rechte der Unternehmung, ist aber auch gleichzeitig Träger aller Pflichten. Der Einzelunternehmer muss für das Eigenkapital selbst aufkommen, und auch das Risiko dessen Verlustes trägt er selbst. Das Vermögen des Inhabers einer Einzelunternehmung spiegelt die Eigenkapitalbasis und somit die Kapitalkraft wieder, wird aber dadurch auch begrenzt. Darüber hinaus haftet der Einzelunternehmer mit seinem gesamten Geschäfts- und Privatvermögen.

### Gesellschaftsunternehmen - Personengesellschaft

Ein wesentliches Merkmal der Personengesellschaft ist, dass mehre Teilhaber sich die Rechte und Pflichten, welche im HGB bzw. im Gesellschaftervertrag geregelt sind, teilen. Das Kapital ist von mehreren Personen aufzubringen. Dadurch teilt sich aber auch die Haftung und die Verantwortung auf diese Personen auf.

Durch die zunehmende Kapitalbasis erhöht sich die Kreditwürdigkeit der Unternehmung. Und nicht zuletzt ist die im Vordergrund stehende persönliche Mitarbeit der Inhaber ein Kennzeichnen für eine Personengesellschaft. Unter dem Sammelbegriff Personengesellschaften sind folgende Unternehmensformen zu finden:

- Offene Handelsgesellschaft (OHG)
- Kommanditgesellschaft (KG)
- GmbH & Co. KG
- Stille Gesellschaft
- Gesellschaft bürgerlichen Rechts (GbR)

### OHG - Offene Handelsgesellschaft

Mindestens zwei Personen müssen sich zusammenschließen und einen Betrieb eines vollkaufmännischen Handelsgewerbes in Form einer Firma ausführen. Plichten und Rechte teilen sich gleichermaßen unter den Gesellschaftern auf. Bei der OHG bestehen keine Vorschriften über die Höhe der Einlagen und des Kapitals. Im Vordergrund steht die Mitarbeit der Gesellschafter. Durch die unbeschränkte Haftung besteht eine hohe Kreditwürdigkeit.

Alle Gesellschafter der OHG haften unbeschränkt, unmittelbar und solidarisch. Unbeschränkt, weil alle Gesellschafter mit dem gesamten Geschäfts- und Privatvermögen haften, unmittelbar, weil Gläubiger nicht zuerst Ansprüche gegenüber der OHG richten müssen, sondern direkt an die Gesellschafter herantreten können und solidarisch, auch gesamtschuldnerisch genannt, weil Gläubiger sich einen Gesellschafter aussuchen können, der dann für die Gesamtschuld aufkommen muss.

### KG - Kommanditgesellschaft

Der Zweck einer KG ist der Betrieb eines Handelsgewerbes unter gemeinschaftlicher Firma und besteht aus zwei Arten von Gesellschaftern, dem Komplementär, welcher Vollhafter ist und dem Kommanditist, welcher Teilhafter ist. Die KG eignet sich besonders für Familiengesellschaften, wo zum Beispiel der Vater der Komplemantär und die Kinder Kommanditisten sein könnten. Eine Erhöhung des Geschäftkapitals kann durch Aufnahme von weiteren Kommanditisten erreicht werden. Der Vorteil dabei ist, dass die Geschäftsführerbefugnis des Komplemantärs davon nicht berührt wird.

Komplementäre haften wie die Gesellschafter der OHG unbeschränkt, unmittelbar und solidarisch. Kommanditisten unterliegen einer beschränkten Haftung, da sie nur bis zu der Höhe ihrer Kapitaleinlage haftbar gemacht werden können.

**GmbH und Co. KG**

Bei einer GmbH & Co. KG ist die GmbH einziger Komplementär einer Kommanditgesellschaft. In der Regel sind die Gesellschafter der GmbH gleichzeitig die Kommanditisten der KG.

Vor allem die Haftungsbeschränkung spricht für diese Unternehmensform. Die GmbH haftet als Vollhafter nur mit dem Gesellschaftsvermögen. Die Teilhafter können nur bis zur Höhe ihrer Kapitaleinlage haftbar gemacht werden.

**GbR - Gesellschaft bürgerlichen Rechts**

Die Gesellschaft des bürgerlichen Rechts ist eine BGB-Gesellschaft. Sie zeichnet sich durch einen dauerhaften oder vorübergehenden Zusammenschluss von mehreren Personen aus. Diese Personen haben ein gemeinsames Ziel.

Die Bildung einer GbR bietet sich an, wenn mit hohem Kapitaleinsatz eines Geschäftes gerechnet werden muss oder das Risiko eines Geschäftes für eine einzelne Person zu groß scheint. Die Gesellschaft des bürgerlichen Rechts wird nicht ins Handelsregister eingetragen. Es entsteht keine Firma aus ihr.

Die Freie und Hansestadt Hamburg (niederdeutsch Friee un Hansestadt Hamborg [ˈhambɔːχ], Abkürzung: HH oder FHH) ist als Stadtstaat ein Land der Bundesrepublik Deutschland. Hamburg ist mit 1,75 Millionen Einwohnern die zweitgrößte Kommune Deutschlands, die drittgrößte deutschsprachige Stadt hinter Berlin und Wien sowie die achtgrößte der Europäischen Union und auch größte Stadt, die nicht Hauptstadt eines ihrer Mitgliedsstaaten ist. Hamburg gliedert sich in sieben Bezirke. Die Stadt bildet das Zentrum der fünf Millionen Einwohnern zählenden Metropolregion Hamburg.

Der Hamburger Hafen ist der größte Seehafen Deutschlands und unter den zwanzig größten Containerhäfen weltweit. Zudem ist Hamburg seit 1996 Sitz des Internationalen Seegerichtshofs (ISGH).

Die älteste urkundliche Erwähnung datiert aus dem 7. Jahrhundert. Durch seinen Vertrag mit Lübeck im Jahr 1241 wurde Hamburg einer der Gründungsorte der Hanse. Hamburg ist Industrie- und Handelsstandort. Die wirtschaftliche Bedeutung der Stadt zeigt sich in der Metropolregion Hamburg, einer der insgesamt elf europäischen Metropolregionen in Deutschland, der Stellung des Hafens, als der zweitgrößte in Europa und vierzehntgrößte weltweit (Stand 2011) und als einer der wichtigsten Medienstandorte Deutschlands.

Mit mehr als 111 Millionen Tagesbesuchern, über 5 Millionen Gästen und über 9,5 Millionen Übernachtungen jährlich ist Hamburg eines der attraktivsten Tourismusziele in Deutschland. Zu den Zielen der Besucher gehören die Hamburger Innenstadt samt Binnenalster, der Hamburger Hafen mit den St. Pauli-Landungsbrücken und der modernen HafenCity samt der Elbphilharmonie, St. Pauli mit der „sündigen Meile“ Reeperbahn und die bekannten Hamburger Bauwerke wie das historische Wahrzeichen Michel. Darüber hinaus sind temporäre Veranstaltungen wie der Hafengeburtstag, der Altonaer Fischmarkt, der Hamburger Dom und der Schlagermove Anziehungspunkte. Am weltweit bedeutenden Musicalstandort Hamburg werden Musicals wie beispielsweise Der König der Löwen aufgeführt.

Hamburg hat über 60 Theater, mehr als 60 Museen und international bekannte Galerien wie die Hamburger Kunsthalle und das Bucerius Kunst Forum. Bedeutende Messen wie die hanseboot oder die Internorga finden regelmäßig statt. Hamburg gilt als Sportstadt, weil neben den Fußballspielen des Hamburger SV und des FC St. Pauli, den Handballspielen des HSV Hamburg, den Radrennen der Vattenfall Cyclassics, den internationalen deutschen Meisterschaften im Tennis auch das deutsche Spring-Derby ausgetragen wird und jährlich der Hamburg-Marathon stattfindet.

Hamburg liegt in Norddeutschland an den Mündungen der Bille und der Alster in die Unterelbe, die etwa 100 km weiter nordwestlich in die Nordsee mündet. An der Elbe erstreckt sich der Tidehafen etwa von der Veddel bis Finkenwerder, hauptsächlich auf dem Südufer der Norderelbe, gegenüber den Stadtteilen St. Pauli und Altona. Die beiden Ufer sind durch die Elbbrücken im Osten sowie durch den Alten und Neuen Elbtunnel verbunden. Das Land südlich und nördlich des Flusses ist Geest, höher gelegene Flächen, die durch die Sand- und Geröllablagerungen der Gletscher während der Eiszeiten entstanden sind. Die unmittelbar am Fluss liegenden Marschen wurden auf beiden Seiten der Elbe über Jahrhunderte von Nebenarmen der Elbe durchzogen und vom Flutwasser der Nordsee überschwemmt, wobei sich Sand und Schlick abgelagert haben. Inzwischen ist die Elbe beidseitig eingedeicht, Nebenarme wurden trockengelegt, umgeleitet, kanalisiert oder abgedeicht. Alte Deichanlagen erinnern in den Außenorten noch an die Zeit, als bei Hochwasser ganze Viertel unter Wasser standen. Höchste Erhebung ist mit 116,2 m ü. NN der Hasselbrack in einem Nordausläufer der Harburger Berge.

Die Alster wird in der Innenstadt zu einem künstlichen See aufgestaut. Dieser teilt sich in die größere Außenalster und die kleinere, vom historischen Kern der Stadt umschlossene Binnenalster. Die Zuflüsse zur Alster wie die Alster selbst sind im Stadtgebiet zum Teil kanalisiert. Sie sind zumeist von ausgedehnten öffentlichen Parkanlagen gesäumt. Die zahlreichen Fleete, Flüsschen und Kanäle der Stadt werden von mehr als 2500 Brücken überspannt. Weithin unbekannt ist, dass sich auf der größten Flussinsel der Elbe, in Wilhelmsburg, einer der letzten Tideauenwälder Europas befindet.

Hamburg grenzt im Norden an Schleswig-Holstein und im Süden an Niedersachsen. Bis auf einige kleinere „Gebietsbereinigungen“, wie den Erwerb der Insel Neuwerk und Flurstücke beim Stauwerk Geesthacht, bestehen die heutigen Grenzen der Stadt Hamburg seit dem Groß-Hamburg-Gesetz, das am 1. April 1937 in Kraft trat. Die Stadt ist nach Berlin sowohl hinsichtlich ihrer Einwohnerzahl als auch ihrer Fläche die zweitgrößte Stadt Deutschlands.

Südlich der Binnenalster liegt das historische Zentrum der Stadt. Der geographische Mittelpunkt von Hamburg in seinen gegenwärtigen politischen Grenzen soll ein Ort am Kuhmühlenteich im Stadtteil Uhlenhorst sein.

Fließgewässer in Hamburg

- Elbe (Unterelbe) mit Norderelbe, Süderelbe, Köhlbrand, Reiherstieg, Rethe, Dove Elbe und Gose Elbe
- Nebenflüsse der Elbe: Seevekanal, Bille, Alster, Flottbek und Este
- Nebenflüsse der Alster: Eilbek (Wandse), Osterbek, Goldbek, Isebek, Tarpenbek, Saselbek, Rodenbek, Bredenbek,

Die Verfassung der Freien und Hansestadt Hamburg legt fest, dass Bezirksämter zu bilden sind. Die Stadt ist verwaltungstechnisch in sieben Bezirke aufgeteilt. Jeder Bezirk gliedert sich in mehrere Stadtteile, von denen es in ganz Hamburg insgesamt 104 gibt; außerdem hat die Stadt 181 Ortsteile. Einige Stadtteile im Kernbereich des Bezirks wurden bis 2008 direkt vom betreffenden Bezirksamt verwaltet, für die anderen Stadtteile des Bezirks gab es jeweils ein eigenes Ortsamt. Insgesamt waren 13 Ortsämter eingerichtet. Anfang 2008 wurden durch eine Gebietsreform die Grenzen einzelner Stadtteile und Bezirke neu gezogen. So fiel der Stadtteil Wilhelmsburg vom Bezirk Harburg an Mitte, und die Stadtteile Sternschanze im Bezirk Altona und HafenCity im Bezirk Hamburg-Mitte wurden neu geschaffen.

Hamburg liegt in der warmgemäßigten Klimazone (effektive Klimaklassifikation nach Köppen und Geiger: Cfb). Aufgrund der durch vorherrschende Westwinde maritimen Einflüsse ist das Klima im Winter milder, im Sommer kühler als im östlichen Hinterland.

Der wärmste Monat ist der Juli mit durchschnittlich 17,4 °C, der kälteste der Januar mit 1,3 °C. Temperaturen um die 28 °C sind im Hochsommer keine Seltenheit. An der Wetterstation Hamburg-Fuhlsbüttel wurde ein Maximalwert von 37,3 °C (9. August 1992) gemessen. Das Klima ist ganzjährig feucht. Im Laufe eines Jahres fallen durchschnittlich 773 mm Niederschlag, an durchschnittlich 52 Tagen im Jahr herrscht Nebel. Im Winterhalbjahr kann es sehr stürmisch werden. Sprichwörtlich ist das Hamburger Schmuddelwetter.

Die ältesten festen Behausungen datieren auf das 4. Jahrhundert v. Chr. für die Ortschaft, die von dem antiken Wissenschaftler Claudius Ptolemäus noch als Treva bezeichnet wurde. Vom 4. bis ins 6. Jahrhundert siedelten sich Sachsen im nordelbischen Raum an.

Im Jahre 810 ließ Karl der Große eine Taufkirche errichten, um den heidnischen Norden zu missionieren. Zur Sicherung der Missionare wurde das Kastell Hammaburg gebaut. 831 begründete Ludwig der Fromme hier ein Bistum, das kurze Zeit später zum Erzbistum wurde. Doch schon kurz nach der Reichsteilung von Verdun überfielen Wikinger die Region, später die slawischen Abodriten, der Erzbischof verlegte seinen Amtssitz nach Bremen. 845 überfielen aus Dänemark stammende Wikinger Hamburg und plünderten die Stadt.

Graf Adolf III. von Schauenburg und Holstein war im 12. Jahrhundert der Gründer einer Handels- und Marktsiedlung am westlichen Alsterufer. Maßgeblich durch das von Kaiser Friedrich I. Barbarossa 1189 verliehene Hafenrecht an diese Siedlung und die Handelsprivilegien für die ganze Unterelbe entwickelte sich die Stadt im Mittelalter zu einem florierenden Handelszentrum und galt mit ihren zeitweilig 600 Brauereien als „Brauhaus der Hanse“. Im 14. Jahrhundert entwickelte sich Hamburg als eines der ersten Mitglieder des Kaufmannsbundes Hanse zum wichtigsten deutschen Umschlag- und Stapelplatz zwischen Nord- und Ostsee. Ab 1510 galt Hamburg endgültig als Reichsstadt. 1558 wurde die Hamburger Börse als eine der ersten Deutschlands eröffnet, im Jahre 1678 unter dem Namen Opern-Theatrum die erste deutsche Oper am Gänsemarkt. Zur Reformationszeit wurde der Stadtstaat ohne Blutvergießen evangelisch. Die Stadt Hamburg erlebte ihre kulturelle Blüte vor allem im 17. und 18. Jahrhundert unter anderem mit der Gründung des Hamburgischen Nationaltheaters (1767).

Auch nach dem Niedergang der Hanse und während der Aufklärung und der Industrialisierung blieb die Stadt neben Berlin das bedeutendste Wirtschaftszentrum Norddeutschlands. Hamburg blieb von den Auswirkungen des Dreißigjährigen Krieges verschont und konnte diesen zum Vorteil nutzen, um seine Vormachtstellung im Handel auszubauen. In ihrer wechselvollen Geschichte unterstand die Stadt der dänischen Königskrone (aber nie von Hamburg formal anerkannt), war Teil des Heiligen Römischen Reiches deutscher Nation und Hauptstadt des Départements Elbmündung (Département des Bouches de l'Elbe) im französischen Kaiserreich (Hamburger Franzosenzeit). 1813–1814 wurde Hamburg vom russischen General Bennigsen belagert. Als Freie Stadt trat es 1815 nach dem Wiener Kongress dem Deutschen Bund bei. 1867 wurde es Mitglied des von Otto von Bismarck initiierten Norddeutschen Bundes und blieb 1871 Gliedstaat des nun in Deutsches Reich umbenannten Bundesstaates.

Besondere Ereignisse der Neuzeit waren der große Hamburger Brand 1842, die Choleraepidemie 1892, der erhebliche Flächen- und Bevölkerungszuwachs 1937/38 durch das Groß-Hamburg-Gesetz, die Bombardierungen im Zweiten Weltkrieg 1943, die Zerstörung der jüdischen Gemeinde (→ Geschichte der Juden in Hamburg), die Errichtung des Konzentrationslagers Neuengamme und seiner zahlreichen Nebenlager im Stadtgebiet, die Sturmflut 1962, die Anbindung an das internationale Straßennetz und den Flugverkehr (Finkenwerder und Fuhlsbüttel), die Veränderung im Hafen und die Auseinandersetzungen um die Hafenstraße in den 1980er-Jahren.

Hamburgs Politik war immer auf größtmögliche Freiheit ihres Handels und politische Unabhängigkeit ausgerichtet. Auch heute noch ist Hamburg als Stadtstaat weitgehend selbständig und bietet dem Handel mit dem größten deutschen Seehafen gute Voraussetzungen.

## Практическое занятие № 9. КОНКУРС ПРОЕКТОВ «РАБОТА АКЦИОНЕРНОГО ОБЩЕСТВА»

**Цель работы**: Усвоение лексики по теме: «предприятие», «формы предприятий Германии»; формирование умений построения монологического текста-описания, а также поисковых и аналитических умений; получение страноведческих знаний.

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

Проект. Все студенты заранее получили задание подготовить сообщение об одном из AG Германии, включающем письменный текст, устный доклад и инфографику. На занятии каждый представляет свой проект и отвечает на вопросы студентов и преподавателя.

### AG - Aktiengesellschaft

Die Aktiengesellschaft verfügt über ein zerlegtes Grundkapital. Man spricht hierbei von den Aktien. Jeder Aktienbesitzter ist Teilhaber der Gesellschaft, hat aber weder Geschäfts- noch Vertretungsbefugnis. Die Teilhaber der Aktiengesellschaft haften nur mit Einlage, also dem Wert der Aktie oder der Aktien. Die Aktiengesellschaft besitzt drei Organe, den Vorstand, den Aufsichtsrat und die Hauptversammlung.

### Gründungsvoraussetzungen

Bei der Gründung einer Aktiengesellschaft sind mindestens 5 Personen erforderlich. Das Grundkapital muss einen Wert von mindestens 50000 Euro betragen, wobei es egal ist, ob es sich um eine Bar- oder Sachgründung handelt. Die Satzung einer AG muss notariell beurkundet und die Gesellschaft in dem Handelsregister eingetragen sein.

### Rechte und Pflichten der Teilhaber einer AG

Aktionäre haben das Recht auf Auskunft durch eine Hauptversammlung. Gemessen an den Anteilen, das heißt den Besitz von Aktien, haben sie ebenso ein Stimmrecht auf der Versammlung. Aktionäre haben das Recht auf Anteil des Gewinnes, der Dividende, und bei der Ausgabe von neuen Aktien besitzen sie ein Bezugsrecht.

Bei einer Auflösung der AG haben sie das Rechte auf Anteil des Gesellschaftsvermögens, aber erst nachdem alle Gläubiger bedient wurden. Die Pflichten eines Aktionärs beschränken sich auf die Einlagen von Kapital, in Höhe des Aktienwertes und der Haftbarkeit durch den Besitz der Aktien.

### Aufgabe der Organe einer AG

Der Vorstand hat die Aufgabe der Geschäftsführung und der Vertretung. Dabei unterliegt er der Sorgfalts- und Haftpflicht.

Der Aufsichtsrat wählt und überwacht den Vorstand und dessen Geschäftsführung. Weitere Aufgaben des Aufsichtsrates sind die Prüfung der Jahresabschlüsse und des Lageberichts und Erstellung eines Berichtes über die Jahresabschlüsse und Lageberichte für die Hauptversammlung. Vorschläge für die Gewinnverteilung zu unterbreiten, gehören ebenso zu seinen Aufgaben.

Die Hauptversammlung entscheidet über den Verwendungvorgang des Bilanzgewinnes. Beschlüsse werden über die Dreiviertelmehrheit entschieden. Des Weiteren bestellt die Hauptversammlung die Abschlussprüfer und die von den Anteilseignern zu wählenden Aufsichtsratsmitglieder.

### Wirtschaftliche Bedeutung einer AG

Meist können größere wirtschaftliche Vorhaben realisiert werden, da das Grundkapital von mehreren Kapitalgebern bereit gestellt wird. Die Aktie dient oft als Anlagemittel, welches den Vorteil der leichten Veräußerung hat.

Großunternehmen greifen meistens auf diese Form der Unternehmung zurück und erreichen durch eine relativ geringe Kapitalbeteiligung eine breite Streuung des Produktionsvermögens.

### GmbH - Gesellschaft mit beschränkter Haftung

Die Gesellschaft mit beschränkter Haftung kann zu jedem gesetzlichem Zweck errichtet werden, dabei spielt die Personenanzahl eine untergeordnete Rolle. Es ist somit auch eine Ein-Mann-GmbH möglich.

Die Gesellschafter haften nur mit der Höhe ihrer Einlage. Die GmbH muss über Stammkapital verfügen, welches wenigstens 25.000 Euro sein muss, eine Stammeinlage eine Höhe von 500 Euro. Ein notariell beglaubigter Gesellschaftsvertrag ist Pflicht, sowie die Eintragung in das Handelsregister.

### Rechte und Pflichten der GmbH-Gesellschafter

Die Gesellschafter einer GmbH haben des Recht auf Anteile des Gewinnes. Die Anteile stehen immer im Verhältnis zu den Geschäftsanteilen, es sei den die Satzung sieht etwas anderes vor. Ebenfalls richtet sich nach den Geschäftsanteilen das Stimmrecht in der Gesellschafterversammlung. Des Weiteren besteht das Recht an Liquidationserlösen.

Die Gesellschafter haben die Pflicht der Einlage des Stammkapitals und der Verlustbeteiligung. Sie haben beschränkte Haftpflicht und Nachschußpflicht, falls dies un der satzung festgehalten ist.

### Aufgaben der Organe einer GmbH

Die Organe der GmbH sind die Geschäftsführer, der Aufsichtsrat und die Gesellschaftsversammlung. Die Aufgaben der Geschäftsführer sind die Geschäftsführung nach innen und die Vertretung nach außen.

Bei Gesellschaften über 500 Arbeitgebern ist die Bildung eines Aufsichtsrates gesetzlich vorgeschrieben. Die Aufgaben des Aufsichtsrates bestehen in der Überwachung der Geschäftführung und die Prüfung des Jahresabschlusses. Darüber hinaus erstellt der Aufsichtsrat den Bericht über den Jahresabschluss für die Gesellschaftsversammlung.

Durch die Gesellschafterversammlung werden Jahresbilanz und Verteilung des Reingewinnes festgestellt. Sie ist für die Bestellung von Geschäftsführern zuständig, aber auch für die Abberufung. Die Gesellschafterversammlung überwacht die Geschäftsführung und bestellt den Prokuristen und die Generalhandlungsbevollmächtigten.

### Genossenschaft

Mittels eines gemeinschaftlichen Geschäftbetriebes fördert die Genossenschaft den Erwerb oder die Wirtschaft der einzelnen Mitglieder. Genossenschaften bestehen aus einer nicht geschlossenen Mitgliederzahl, welche auch Genossen genannt werden. Nur das Vermögen der Genossenschaft ist haftbar zu machen.

Zur Bildung einer Genossenschaft müssen mindestens sieben Mitglieder bestehen. Die Genossenschaft muss einen schriftlichen Gesellschaftsvertrag aufsetzen. Des weiteren sind Genossenschaften ins Genossenschaftenregister einzutragen. Daraus ergibt sich eine konstitutive Wirkung des Formkaufmanns.

Die Rechte und Pflichten der Genossen werden im Gesellschaftsvertrag festgelegt. Die Genossen haben Stimmrecht und das Recht auf Gewinnbeteiligung. Sie haben Kündigungsrecht, sind aber zur Einlage und Nachschuß verpflichtet.

**Aufgaben der Organe einer Genossenschaft**

Genossenschaften bestehen dem Vorstand, dem Aufsichtsrat und der Generalversammlung. Der Vorstand, welcher sich aus mindestens zwei von der Generalversammlung gewählten Genossen besteht, hat Geschäftsführung- und Vertretungsbefugnis.

Der Aufsichtsrat hat die gleichen Rechte wie die Aufsichtsratmitglieder der AG. Der Aufsichtsrat bei einer Genossenschaft muss aus wenigstens drei Genossen bestehen.

Die Generalversammlung ist das oberste Entscheidungsorgan und beschließt über die Gewinnbeteiligung und die Führung der Geschäfte. Die Rechte entsprechen denen der Hauptversammlung einer AG.

**Vorschriften für Firmennamen**

Der Einzelkaufmann hat seinen Familiennamen mit ausgeschriebenen Vornamen als Firma zu führen. Zusätze sind erlaubt.

Die OHG erhält die Zunamen aller Teilhaber, oder den Namen eines Teilhabers mit Zusatz, welcher das Gesellschaftsverhältnis andeutet.

Die KG muss den Namen von mindestens einem Vollhafter mit dem Zusatz KG führen. Teilhafter dürfen nicht namentlich aufgenommen werden.

Bei einer GmbH die als Personenfirma geführt wird, ist der Name von wenigstens einem Gesellschafter mit Zusatz zu führen, bei einer GmbH als Sachfirma die Bezeichnung der Unternehmung mit Zusatz.

Die Genossenschaft wird als Sachfirma mit dem Zusatz eG geführt.

## Практическое занятие № 10. ДЕЛОВАЯ ИГРА «ДЕЛОВЫЕ ПИСЬМА»

***Цель работы***: *Усвоение структуры и лексики делового письма, видов делового письма; формирование умений построения писем различных видов; получение страноведческих знаний.*

Для выполнения практического занятия обучающимся необходимы тексты для анализа и словари (http://www.babla.ru/).

*Игра 1. Студенты делятся на 2-3 группы. Каждая группа готовит презентацию своей фирмы (название, виды деятельности). Затем группы начинают обмениваться письмами. Виды писем определяются преподавателем.*

| | | | |
|---|---|---|---|
| **Helmut Wagner & Sohn**<br>**Kältetechnik**<br><br>Helmut Wagner & Sohn, Postfach 256, 3500 Kassel | | | |
| Schrader & Lehmann<br>Einkaufsabteilung<br>Max-Richter-Strasse 95<br>8770 Potsdam | | | |
| | | | |

| Ihre Zeichen, Ihre Nachricht vom 02.04.2004 | Unsere Zeichen, unsere Nachricht vom 04.04.2004 | (05 61) 8243-1 Durchwahl 8243 | Kassel 08.04.2004 |
|---|---|---|---|
| | | | |

Rückfrage
Sehr geehrte Damen und Herren,
bezugnehmend auf Ihre Bestellung über eine Kühlanlage müssen wir Ihnen folgendes mitteilen: Es stellte sich heraus, dass bei der forgegebenen Grösse des Kühlraums ein stärkeres Kühlaggregat eingebaut werden muss, was eie Verteuerung des Preises um 8% hervorruft. Nun möchten wir uns erkundugen, ob Sie mit dieser Verteuerung einverstanden sind.
Bitte, teilen Sie uns Ihren Entscheid mit.
Mit freundlichen Grüssen
(Unterschrift)
Helmut Wagner

**Henneberg & Co**
Schwarzwalder

Holzwarenindustrie Nagold

Herbert Henneberg & Co. Postfach 23.7270 Nagold

Spielwarenhandlung
Karl Reinhardt
Bremer Strasse 28
2000 Hamburg 12

| Ihre Zeichen, Ihre Nachricht vom 03.05.2005 | Unsere Zeichen, unser Nachricht vom 28.04.2005 | (07452) 4288 Hausapparat | Nagold 10.05.2005 |
|---|---|---|---|
| | | | |

Angebot uber Spielwaren

Sehr geehrter Herr Reinhardt,
wir freuen uns, dass Sie Interesse an unseren Holzspielwaren haben und senden Ihnen gern den gewunschten Katalog mit der neuesten Preisliste.
Beachten Sie bitte unsere gunstige Liefer- und Zahlungsbedingungen am Ende des Katalogs.
Wir hoffen, dass unsere Holzspielwaren Ihren Verkaufsvorstellungen entsprechen und wir Sie bald beliefern konnen.

Mit freundlichen Grussen
Herbert Henneberg & Co.
ppa.
(Unterschrift)
Anette Prollius

**Firma A. Koch Wallstr. 12. 3150 Peine**

| Ihre Zeichen Ihre Nachricht vom 03.12.2004 | Unsere Zeichen vom 08.12.2004 | Telefon (0 53 21) 2 25 78 | Coslar 05.12.2004 |
|---|---|---|---|
| | | | |

Lieferanzeige

Sehr geehrte Damen und Herren,
Als Frachtgut senden wir Ihnen heute eine Teillieferung von 500 St. Nr. 43/75. Den Rest von 300 St. werden wir Mitte Januar liefern.

Mit fruendlichen Grüssen

Emil Otto & Co.

(Unterschrift)

**Bankhaus**
Friederich

Bauer

Bankhaus F. Bauer AG., Postfach 294, 6100 Darmstadt

Gerb. Winkelmann
Werbemittel
Rothschildallee 104
600 Frankfurt 1

| Ihre Zeichen, Ihre Nachricht vom 04.10.04 | Unsere Zeichen, unsere Nachricht vom 01.10.04 | (0 6165) 1425-1 Durchwahl 1425124 | Darmstadt 15.10.05 |
|---|---|---|---|

Bestellung

Sehr geehrte Damen und Herren,
wir danken Ihnen für Ihr Angebot. Endsprechend Ihrer Mustersendung bestellen wir:

200 Brieftaschen, Nr. 5714, schwarz, mit Prägedruck auf der linken Innenseite:

Bankhaus Friedrich Bauer AG,
Preis __,__ EU je Stück,

200 Geldbörsen, Nr 3272, schwarz, mit Prägedruck wie oben auf der linken Innenseite,
Preis __,__ EU je Stück.

Liefern Sie bitte binnen vier Wochen frei Haus. Bei Bezahlung innerhalb zweier Wochen nach Wareneingang ziehen wir 2% Skonto vom Warenwert ab.

Mit freundlichen Grüßen
Bankhaus Friedrich Bauer AG
ppa. (Unterschrift)
Inge Weber

Telefax
An: Hrn. W. von Rhein
Fa: von Rhein Arzneimittel GmbH
Fax: 0102/334422
Von: Mann-Computer GmbH
Fax: 090/364704
Datum: 19.03.2004
Seiten: 1

Sehr geehrter Herr von Rhein,

Wir danken Ihnen fur Auftrag uber Schreibautomaten und bestatigen hiermit, dass wir gemass unseren allgemeinen Verkaufsbedingungen liefern konnen:

20 Schreibautomaten Modell X2AL zu einem Preis von EU 590,-/Stck inkl. Verpackung.

Lieftermin: 11 bis 20 Tagen

Lieferung: frei Ihrer Lubecker Fabrik
Zahlung: innerhalb 30 Tage
Wegen naherer Einzelheiten setzen wir uns mit Ihnen wieder in Verbindung.

Mit freundlichen Grussen,

Mann-Computer GmbH

**Литература**

**Основная учебная**


1. **Хачатурьян, К.Г.** Учебное пособие по немецкому языку [Электронный ресурс]: учебное пособие/ Хачатурьян К.Г. – Электрон. текстовые данные. – Краснодар: Южный институт менеджмента, 2012. – 29 c. – http://www.iprbookshop.ru/9574. – ЭБС «IPRbooks».
2. **Шишонина, Н.В.** Немецкий язык. Введение. Фонетика. Морфология. Синтаксис. Бытовая и общеупотребительная лексика. Профессионально-ориентированные тексты. Тренировочные упражнения [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2011. - http://lib.muh/ru.
3. **Шишонина, Н.В.** Немецкий язык. Морфология. Синтаксис. Бытовая и общеупотребительная лексика. Профессионально-ориентированные тексты. Тренировочные упражнения [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2012. - http://lib.muh/ru.
4. **Шишонина, Н.В.** Немецкий язык. Морфология. Сложные грамматические конструкции. Часть 1 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2013. - http://lib.muh/ru.
5. **Шишонина, Н.В.** Деловой курс немецкого языка. Часть 1 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2013. - http://lib.muh/ru.
6. **Шишонина, Н.В.** Деловой курс немецкого языка - 3 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2012. - http://lib.muh/ru..
7. **Шишонина, Н.В.** Деловой курс немецкого языка - 4 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2012. - http://lib.muh/ru..


**Дополнительная**


1. **Санарова, Е.Г.** Немецкий язык для Вас [Электронный ресурс]: учебное пособие/ Санарова Е.Г. – Электрон. текстовые данные. – Краснодар: Южный институт менеджмента, 2012. – 75 c. – http://www.iprbookshop.ru/9775. – ЭБС «IPRbooks».
2. **Санарова, Е.Г.** Немецкий язык для Вас [Электронный ресурс]: учебное пособие/ Санарова Е.Г. – Электрон. текстовые данные. – Краснодар: Южный институт менеджмента, 2012. – 84 c. – http://www.iprbookshop.ru/9776. – ЭБС «IPRbooks».
3. **Яворская, И.Б.** Практикум по грамматике немецкого языка [Электронный ресурс]: учебное пособие/ Яворская И.Б. – Электрон. текстовые данные. – Краснодар: Южный институт менеджмента, 2010. – 35 c. – http://www.iprbookshop.ru/9794. – ЭБС «IPRbooks».
4. **Шишонина, Н.В.** Немецкий язык. Морфология. Сложные грамматические конструкции. Часть 2 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2013. - http://lib.muh.ru.
5. **Шишонина, Н.В.** Деловой курс немецкого языка. Часть 2 [Электронный ресурс]: рабочий учебник/Шишонина Н.В. - 2013. - http://lib.muh/ru.


**Материально-техническое обеспечение**

Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине представлено в приложении 7 «Сведения о материально-техническом обеспечении программы высшего образования – программы бакалавриата направления подготовки 09.03.01 «Информатика и вычислительная техника»

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «НЕМЕЦКИЙ ЯЗЫК. БАЗОВЫЙ КУРС ДЛЯ НЕЛИНГВИСТОВ» (КУРС 1)**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Горбатова Н.И.
Оператор компьютерной верстки Белюсенко Е.В.

**5734.01.01;МУ.01;1**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

# ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «ФИЗИЧЕСКАЯ КУЛЬТУРА И СПОРТ»

**МОСКВА 2018**

Разработано И.С. Барчуковым, д.п.н., проф.,
Е.В. Потаповой

Рекомендовано Учебно-методическим советом
в качестве методических указаний для
педагогических работников и обучающихся
образовательной организации

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

# ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ ПО ДИСЦИПЛИНЕ «ФИЗИЧЕСКАЯ КУЛЬТУРА И СПОРТ» (КУРС 1)

Методические указания подготовлены для педагогических работников и обучающихся в образовательной организации, предназначены для подготовки к организации и проведению занятий по дисциплине «Физическая культура и спорт» и являются неотъемлемой частью дидактического обеспечения подготовки бакалавров.

Для педагогических работников и обучающихся

**О Г Л А В Л Е Н И Е**

## 1 ОБЩИЕ ПОЛОЖЕНИЯ

### 1.1 Цели и задачи дисциплины

***Цель дисциплины*** - формирование физической культуры личности и способности направленного использования разнообразных средств физической культуры, спорта для сохранения и укрепления здоровья.

***Задачи дисциплины:***

– знание научно-биологических и практических основ физической культуры и здорового образа жизни;

– формирование мотивационно - ценностного отношения к физической культуре, установки на здоровый стиль жизни, физическое самосовершенствование и самовоспитание, потребности в регулярных занятиях физическими упражнениями и спортом;

– овладение системой практических умений и навыков, обеспечивающих сохранение и укрепление здоровья, психическое благополучие, развитие и совершенствование психофизических способностей, качеств и свойств личности, самоопределение в физической культуре;

– приобретение опыта творческого использования физкультурно-спортивной деятельности для достижения жизненных и профессиональных целей;

– понимание социальной роли физической культуры в развитии личности и подготовке ее к профессиональной деятельности.

### 1.2 Планируемые результаты обучения по дисциплине

В результате изучения дисциплины обучающийся должен:

***знать:***

- роль физической культуры в развитии человека и подготовке бакалавра;
- сущность физической культуры в общекультурной и профессиональной подготовке обучающихся;
- социально-биологические основы физической культуры и спорта;
- основы здорового образа жизни обучающихся;
- особенности использования средств и методов физической культуры для оптимизации работоспособности;
- общую физическую подготовку обучающихся в системе физического воспитания.

***уметь:***

- индивидуально выбирать вид спорта или систему физических упражнений для своего физического совершенствования;
- применять на практике физическую подготовку обучающихся.

***владеть****:*

- личным опытом использования физкультурно-спортивной деятельности для повышения своих функциональных и двигательных возможностей, для достижения личных жизненных и профессиональных целей;
- системой практических умений и навыков, обеспечивающих сохранение и укрепление здоровья, развитие и совершенствование психофизических способностей и качеств (с выполнением установленных нормативов по общей физической подготовке);
- методиками самостоятельных занятий и самоконтроля над состоянием своего организма.

### 1.3 Место дисциплины в структуре образовательной программы

Дисциплина «Физическая культура и спорт» относится к базовой части Блока 1.

**1.4 Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем и на самостоятельную работу обучающихся**

| Виды учебных занятий | Всего часов по формам обучения (в академ. часах) | | |
|---|---|---|---|
| | **Очная** | **Очно-заочная** | **Заочная** |
| **Контактная работа (объем работы обучающихся во взаимодействии с преподавателем) (всего)** | **32** | **-** | **12** |
| занятия лекционного типа (лекции) | 8 | - | 4 |
| занятия семинарского типа (практические, интерактивные): практические занятия по физической культуре в группе, реферат, асессинг письменной работы, устное эссе, асессинг устного выступления, тест-тренинг, логическая схема, глоссарный тренинг, модульное тестирование и т.д. | 24 | - | 8 |
| из них:<br>- консультации (групповые и индивидуальные-IP-helping) | - | - | - |
| - курсовое проектирование (выполнение курсовой работы) | - | - | - |
| занятия семинарского типа: лабораторные работы (лабораторные практикумы) | - | - | - |
| *Объем занятий, проводимых путем непосредственного взаимодействия педагогического работника с обучающимися* | *14* | | - |
| *Объем занятий с применением электронного обучения, дистанционных образовательных технологий* | *18* | | *12* |
| **Самостоятельная работа (всего)** | **40** | **-** | **56** |
| Работа в электронной информационно-образовательной среде с образовательными ресурсами интегральной учебной библиотеки компьютерными средствами обучения для подготовки к текущей и промежуточной аттестации, к курсовому проектированию (выполнению курсовых работ), в т.ч. консультации (групповые и индивидуальные-IP-helping) | 40 | - | 56 |
| Вид промежуточной аттестации: зачет | 2* | - | 4 |
| **Общая трудоемкость дисциплины** часы | **72** | **-** | **72** |
| зачетные единицы | 2 | | 2 |

** Часы для проведения зачета включены в занятия семинарского типа (практические, интерактивные).*

**1.5 Содержание дисциплины, структурированное по темам (разделам)**

| № п/п | Наименование раздела дисциплины | Содержание тем раздела |
|---|---|---|
| 1 | Физическая культура в общекультурной подготовке обучающихся. Социально-биологические основы физической культуры. Основы здорового образа жизни обучающегося, физическая культура в обеспечении здоровья. Средства физической культуры в регулировании работоспособности. Общая физическая подготовка. Основы методики | **Психофизиологические основы учебного труда и интеллектуальной деятельности, средства физической культуры в регулировании работоспособности** Психофизиологическая характеристика интеллектуальной деятельности и учебного труда студента. Динамика работоспособности студентов в учебном году и факторы, ее определяющие. Основные причины изменения психофизического состояния студентов в период экзаменационной сессии, критерии нервно-эмоционального и психофизического утомления. Особенности использования средств физической культуры для оптимизации работоспособности, профилактики нервно-эмоционального и психофизического утомления студентов, повышения эффективности учебного труда. Методические принципы физического воспитания. Методы |

| № п/п | Наименование раздела дисциплины | Содержание тем раздела |
|---|---|---|
| | самостоятельных занятий физическими упражнениями | физического воспитания. Основы обучения движениям. Основы совершенствования физических качеств. Формирование психических качеств в процессе физического воспитания.<br>Общая физическая подготовка, ее цели и задачи. Спортивная подготовка, ее цели и задачи. Структура подготовленности спортсмена. Зоны и интенсивность физических нагрузок. Значение мышечной релаксации. Возможность и условия коррекции физического развития, телосложения, двигательной и функциональной подготовленности средствами физической культуры и спорта в студенческом возрасте. Формы занятий физическими упражнениями. Учебно-тренировочное занятие как основная форма обучения физическим упражнениям. Структура и направленность учебно-тренировочного занятия.<br>**Основы методики самостоятельных занятий физическими упражнениями**<br>Мотивация и целенаправленность самостоятельных занятий. Формы и содержание самостоятельных занятий. Организация самостоятельных занятий физическими упражнениями различной направленности. Характер содержания занятий в зависимости от возраста. Особенности самостоятельных занятий для женщин. Планирование и управление самостоятельными занятиями. Границы интенсивности нагрузок в условиях самостоятельных занятий у лиц разного возраста. Взаимосвязь между интенсивностью нагрузок и уровнем физической подготовленности. Гигиена самостоятельных занятий. Самоконтроль за эффективностью самостоятельных занятий. Участие в спортивных соревнованиях. |
| 2 | Индивидуальный выбор видов спорта или систем физических упражнений. Самоконтроль занимающихся физическими упражнениями и спортом.<br>Физическая культура в профессиональной деятельности бакалавра. | **Индивидуальный выбор видов спорта или систем физических упражнений**<br>Массовый спорт и спорт высших достижений, их цели и задачи. Спортивная классификация. Студенческий спорт. Особенности организации и планирования спортивной подготовки в образовательной организации. Спортивные соревнования как средство и метод общей физической, профессионально-прикладной, спортивной подготовки студентов. Система студенческих спортивных соревнований. Общественные студенческие спортивные организации. Олимпийские игры и Универсиады. Современные популярные системы физических упражнений. Мотивация и обоснование индивидуального выбора студентом вида спорта или системы физических упражнений для регулярных занятий. Краткая психофизиологическая характеристика основных групп видов спорта и систем физических упражнений.<br>**Особенности занятий избранным видом спорта или системой физических упражнений**<br>Краткая историческая справка. Характеристика особенностей воздействия данного вида спорта (системы физических упражнений) на физическое развитие и подготовленность, психические качества и свойства личности. Модельные характеристики спортсмена высокого класса. Определение цели и задач спортивной подготовки (или занятий системой физических упражнений) в условиях образовательной организации. Возможные формы организации тренировки в образовательной организации.<br>Перспективное, текущее и оперативное планирование подготовки. Основные пути достижения необходимой структуры подготовленности занимающихся. Контроль за эффективностью тренировочных занятий. Календарь студенческих соревнований. Спортивная классификация и правила спортивных соревнований в избранном виде спорта.<br>**Самоконтроль занимающихся физическими упражнениями и спортом** |

| № п/п | Наименование раздела дисциплины | Содержание тем раздела |
|---|---|---|
| | | Диагностика и самодиагностика состояния организма при регулярных занятиях физическими упражнениями и спортом. Врачебный контроль, его содержание. Педагогический контроль, его содержание. Самоконтроль , его основные методы, показатели и дневник самоконтроля. Использование методов стандартов, антропометрических индексов, номограмм функциональных проб, упражнений-тестов для оценки физического развития, телосложения, функционального состояния организма, физической подготовленности. Коррекция содержания и методики занятий физическими упражнениями и спортом по результатам показателей контроля<br><br>**Физическая культура и спорт в профессиональной деятельности бакалавра**<br>Производственная физическая культура. Производственная гимнастика. Особенности выбора форм, методов и средств физической культуры и спорта в рабочее и свободное время специалистов. Профилактика профессиональных заболеваний и травматизма средствами физической культуры. Дополнительные средства повышения общей и профессиональной работоспособности. Влияние индивидуальных особенностей, географо-климатических условий и других факторов на содержание физической культуры специалистов, работающих на производстве. |

Для овладения системой практических умений и навыков, обеспечения общей физической подготовленности обучающихся необходимо проведение практических занятий по физической культуре и спорту.

Для проведения практических занятий по физической культуре (физической подготовке) формируются учебные группы численностью не более 15 человек с учетом пола, состояния здоровья, физического развития и физической подготовленности обучающихся.

***Места проведения практических занятий:*** учебные аудитории образовательной организации и (или) при использовании Roweb-технологии сайт «Личная студия»; при проведении практических занятий – учебная аудитория для проведения занятий по дисциплинам по физической культуре и спорту.

***Оборудование для проведения занятий по физической культуре и спорту:*** Мячи фитболы, мячи набивные, палки гимнастические (деревянные), палки гимнастические (пластик), скакалки гимнастические, коврики гимнастические, степ-доска, обруч металлический, обруч пластиковый, стенка гимнастическая, турник универсальный, доска для пресса, скамья гимнастическая, стойки для штанги, гриф кривой, гриф классический, подставка для жима лежа, гантели разновесы: 5 кг., 3 кг, 2 кг, диски разновесы: 20 кг.,15 кг., 10 кг., 5 кг., гири разновесы: 16 кг , 24 кг. , шахматы, шашки.

## 2 СОДЕРЖАНИЕ УЧЕБНЫХ ЗАНЯТИЙ ПО ФИЗИЧЕСКОЙ КУЛЬТУРЕ И СПОРТУ

Учебные занятия по физической культуре и спорту являются важным компонентом повышения разносторонних физических, психологических и других качеств личности. Они включают в себя приемы и действия из различных разделов физической культуры и следующие ***задачи***:

- эффективно совершенствовать физические, морально-волевые и психологические качества, а также двигательные навыки за счет тренировок, целенаправленного воздействия упражнений, широкого применения различных вариантов нагрузок;
- развивать у обучающихся способность быстро переключаться с одного вида мышечной деятельности на другую;

- значительно увеличивать плотность занятия и развивать способность обучающихся переносить интенсивные физические нагрузки, приближать характер физической культуры к соревновательному режиму и игровой деятельности.

Учебные занятия по физической культуре и спорту различаются по содержанию и направленности и подразделяются на теоретико-практические, практические, а также выполняемые самостоятельно во внеучебное время.

К *теоретико-практическим занятиям* относятся занятия, на которых помимо выполнения практических упражнений отражается содержание разделов программы: роль физической культуры в общекультурном, социальном развитии человека, организация и проведение учебных занятий в образовательной организации и др.

*Практические занятия* включают гимнастику, комплексные занятия, а также занятия в электронной информационно-образовательной среде (реферат, учебное экспертирование реферата, мониторинг работы с текстами, штудирование, модульное тестирование и др.).

*Самостоятельная работа* проводится во внеучебное время и преследует цели совершенствования изученных приемов и действий, а также подготовку к следующим занятиям, подготовку к рефератам.

## 3 ЭТАПЫ ПРОВЕДЕНИЯ УЧЕБНОГО ЗАНЯТИЯ ПО ФИЗИЧЕСКОЙ КУЛЬТУРЕ И СПОРТУ

Учебное практическое занятие состоит из подготовительной, основной и заключительной частей. Подготовительная часть (не менее 15 мин) – организационные элементы, объяснение содержания занятия, формирование осанки, упражнения на общее физическое развитие и укрепление организма, подготовка к перенесению предстоящих нагрузок в основной части. Ходьба и бег проводятся в колонне по одному или по два с различными положениями рук (на поясе, в стороны, к плечам, за голову, вверх, перед грудью, а также со сгибанием, опусканием, размахиванием и др.).

При выполнении несложных упражнений в ходьбе и беге показ и объяснение целесообразно делать, не останавливая группу, двигаясь ей навстречу; можно осуществлять различные перестроения для тренировки внимания.

Общеразвивающие упражнения (ОРУ) проводятся в следующей последовательности: движения для мышц рук и плечевого пояса, туловища, для мышц ног, всего тела и прыжки. Преподаватель выбирает место перед обучающимися на возвышении или так, чтобы он был хорошо виден всем, и осуществляет показ упражнений (зеркальный способ).

В подготовительной части занятия проводятся групповые упражнения со скамьей гимнастической, с гимнастическими палками, скакалками.

Преподаватель обучает правильному дыханию: вдох – при поднимании рук, выпрямлении тела; выдох – во время наклонов, опускания рук. В иных ситуациях дыхание осуществляется произвольно. Однако задерживать его нельзя: вдох и выдох должны быть непроизвольными.

Основная часть занятия выполняет главную функцию, так как именно в ней решаются категории задач физического воспитания (учебно-воспитательные, физического развития). К ним относятся: формирование знаний в области двигательной деятельности; обучение двигательным умениям и навыкам общеобразовательного и спортивного характера; развитие общих и специальных функций опорно-двигательного аппарата, сердечно-сосудистой и дыхательной систем; формирование и поддержание хорошей осанки; закаливание организма; воспитание нравственных, интеллектуальных, волевых и эстетических качеств. Продолжительность основной части составляет 70–75 % времени, отводимого на занятие.

В основной части вначале рекомендуются разучивать новые двигательные действия или их элементы. Закрепление и совершенствование усвоенных ранее навыков проводится в середине или конце основной части занятия. Упражнения, требующие проявления скоростных, скоростно-силовых качеств, тонкой координации движений, выполняют в начале основной части занятия, а упражнения, связанные с силой и выносливостью, – в конце. Состав всех упражнений в основной части занятия должен быть таким, чтобы они оказывали

разностороннее влияние для поддержания эмоционального тонуса и закрепления пройденного на занятии материала.

В *заключительной части* занятия обучающиеся приводят организм в относительно спокойное состояние с помощью медленной ходьбы, упражнений в глубоком дыхании и на расслабление мышц. При подведении итогов педагогический работник оценивает каждого, указывает на положительные или отрицательные проявления отдельных обучаемых, отмечает наиболее активных и целеустремленных. Затем приводятся в порядок места занятий. На заключительную часть отводится до 10-15 мин.

**Примерный план**
**учебного занятия по выполнению гимнастических упражнений**
**(вариант 1)**

**Задачи:** тренировка в выполнении упражнений развития быстроты, ловкости, выносливости, пространственной ориентировки; воспитание коллективизма, настойчивости и инициативы.

**Время:** 90 мин.

**Форма одежды:** спортивная.

**Материальное обеспечение:** палки гимнастические, скакалки гимнастические, гантели.

**Место проведение**: учебная аудитория для проведения занятий по дисциплинам по физической культуре и спорту

| Содержание | Время (мин) | Организационно-методическое указание |
|---|---|---|
| Подготовительная часть – 15 мин | | |
| Построение, объяснение задач и содержания занятия | 3 | Проводить в двух шеренгах |
| Повороты направо, налево и кругом | 3 | Каждый поворот повторить 2 раза |
| Ходьба с движениями рук вверх и вниз-назад | 3 | Проводить в колонне по одному |
| Бег; бег с высоким подниманием коленей | 3 | Движения руками на каждый шаг |
| Комплексы вольных упражнений | 3 | Повторить 2 раза, медленно и с постепенным ускорением |
| Основная часть – 65 мин | | |
| Подъем переворотом на количество раз | 10 | Выполнять по 2 раза всем одновременно (отстающие – с помощью) |
| Упражнения с фитбольным мячом (гимнастическими палками) | 5 | Выполнять по 10 раз всем попеременно. Варианты выполнения упражнения указаны на рисунках 1, 2, 3 |
| Упражнения с гантелями | 10 | Выполнять по 4 раза на разные группы мышц. Варианты выполнения упражнения указаны на рисунке 2 |
| Упражнения с гимнастической скакалкой | 10 | Выполнять по 10 раз (3 подхода). Варианты выполнения упражнения указаны на рисунке 3 |
| Упражнения на гимнастических ковриках | 10 | Выполнять упражнения на растяжку |
| Упражнения на гимнастической стенке | 10 | Выполнять упражнения на пресс |

| Содержание | Время (мин) | Организационно-методическое указание |
|---|---|---|
| Самостоятельная тренировка упражнений | 10 | Упражнения выполняются индивидуально, в группе под наблюдением преподавателем. |
| Заключительная часть – 10 мин | | |
| Ходьба, упражнения на расслабление мышц и в глубоком дыхании | 5 | Выполнить в колонне по одному |
| Подведение итогов занятия | 5 | |

Преподаватель _____

**Примерный план**
**учебного занятия**
**(вариант 2)**

**Задачи:** тренировка в выполнении упражнений для развития быстроты, ловкости, выносливости, пространственной ориентировки; воспитание коллективизма, настойчивости и инициативы.

**Время:** 90 мин.

**Форма одежды:** спортивная.

**Место проведения:** учебная аудитория для проведения занятий по дисциплинам по физической культуре и спорту

**Материальное обеспечение:** скакалки гимнастические.

| **Содержание занятия** | **Время (мин)** | **Организационно-методическое указание** |
|---|---|---|
| **Подготовительная часть – 15 мин** | | |
| Организация обучающихся.<br>Объяснение задач занятия | 5 | Подготовить места к занятиям |
| Ходьба, бег, упражнения для мышц рук, туловища и ног | 5 | Проводить в колонне по одному; упражнения повторить по 5 раз, дистанция 2 м |
| Специальные упражнения на ловкость, координацию движений | 5 | Проводить в учебной аудитории для проведения занятий по дисциплинам по физической культуре и спорту |
| **Основная часть – 65 мин** | | |
| Физические упражнения:<br>- бег с высоким подниманием бедра;<br>- бег с захлестыванием голени;<br>- приседание на двух ногах | 10 | Выполнять в шеренге одновременно всеми, темп высокий; повторить 2-3 раза |
| Прыжки в длину с места, отталкивание двумя ногами | 10 | Обращать внимание на технику выполнения прыжка |
| Упражнение на универсальном турнике | 10 | Упражнения выполняются индивидуально, в группе под наблюдением преподавателя |
| Упражнения: сгибание и разгибание рук в упоре, поднимание туловища из положения лежа в сед, со скакалкой, комплекс вольных упражнений на 16 счетов | 15 | Тренировку проводить самостоятельно; обратить внимание на проявление инициативы и находчивости |
| Самостоятельная тренировка упражнений | | |

| | | |
|---|---|---|
| | 20 | Упражнения выполняются индивидуально, в группе под наблюдением преподавателя. |
| **Заключительная часть – 10 мин** | | |
| Приведение в порядок мест занятий и инвентаря | 3 | |
| Бег, ходьба в медленном темпе и упражнения в глубоком дыхании | 4 | Проводить в колонне по одному |
| Подведение итогов занятия | 3 | Отметить степень выполнения задач, дать задание на самостоятельную подготовку |

Преподаватель _____

## 4 УПРАЖНЕНИЯ ПРИ ПРОВЕДЕНИИ УЧЕБНЫХ ЗАНЯТИЙ ПО ФИЗИЧЕСКОЙ КУЛЬТУРЕ И СПОРТУ

***Упражнения с гимнастической палкой***

Использование гимнастической палки позволяет разнообразить известные упражнения и обладает следующими преимуществами: корректирует осанку, помогая удерживать спину прямой и зафиксировать позвоночник; развивает подвижность в плечевых суставах; способствует более правильному и четкому выполнению упражнений (рисунок 1).

Рисунок 1. Упражнения с гимнастической палкой

Основные положения гимнастической палки:

- палка перед грудью – палка держится согнутыми в локтях руками у верхней линии груди, немного ниже ключиц;
- палка на плечах — палка держится согнутыми в локтях руками и находится за головой, на линии плеч или чуть ниже;
- палка перед собой – палка находится в прямых руках, вытянутых горизонтально перед собой;
- палка вверху — палка находится в прямых руках, вытянутых вертикально вверх;
- палка внизу — палка находится перед собой в прямых руках на уровне бедер;

- палка за спиной между локтями – палка находится за спиной на уровне пояса или чуть выше, удерживается согнутыми в локтях под прямым углом руками (лежит на локтевых сгибах).

***Упражнения с гимнастической палкой в положении стоя:***

- встать, ноги на ширине плеч, палка на плечах. 1 – повернуть корпус влево, таз и ноги остаются на месте; 2 – вернуться в исходное положение; 3 – повернуть корпус влево; 4 – вернуться в исходное положение;
- встать, ноги на ширине плеч, палка вверху. 1 – наклонить корпус вправо; 2 — вернуться в исходное положение; 3 – наклонить корпус влево; 4 – вернуться в исходное положение. Палка движется строго в вертикальной плоскости;
- встать, ноги на ширине плеч, палка перед собой. 1 – правую руку поднять вверх, левую опустить вниз, палка вертикальна; 2 – наклонить корпус влево; 3, 4 – вернуть корпус и палку в исходное положение; 5 – левую руку поднять вверх, правую руку опустить вниз; 6 – наклонить корпус вправо; 7, 8 – вернуть корпус и палку в исходное положение;
- встать, ноги вместе, палка за спиной между локтями. 1 – отвести правую ногу точно в сторону носком в пол; 2 – наклонить корпус вправо к ноге; 3 – выпрямить корпус; 4 – вернуть ногу в исходное положение; 5 – отвести левую ногу в сторону; 6 – наклонить корпус влево к ноге; 7 – выпрямить корпус; 8 – вернуть ногу в исходное положение;
- встать, ноги на ширине плеч, палка на плечах. 1 – наклониться вперед, максимально прогнувшись в спине (в пояснице и грудном отделе), ноги прямые, затылок составляет со спиной одну линию; 2 — поднять руки с палкой вперед и вверх, максимально отводя ее назад; 3 – вернуть палку в исходное положение; 4 – выпрямиться;
- встать, ноги на ширине плеч, палка находится перед грудью. 1 – наклониться к правой ноге, по возможности положить палку на пол перед ней; 2 – вернуться в исходное положение; 3 – наклониться к левой ноге; 4 – вернуться в исходное положение;
- встать, ноги на ширине плеч, палка внизу. 1 – наклониться вперед, спина прямая, взгляд перед собой; 2 – сгибая руки в локтях, поднять палку к груди, свести лопатки, локти направлены в стороны и вверх; 3 – опустить палку вниз; 4 — вернуться в исходное положение;
- поставить палку вертикально на пол перед собой, положить на ее конец руки, одну ладонь на другую, руки прямые. 1 – наклониться вперед, прогнувшись в спине, опираясь на палку; 2, 3 – выполнить пружинистые покачивания вверх-вниз, увеличивая прогиб; 4 — вернуться в исходное положение.

***Упражнения с гимнастической палкой в положении сидя:***

- сесть, прямые ноги на ширине плеч, палка на плечах. 1 – наклонить корпус вперед, стараясь удерживать спину прямой; 2, 3 – делать пружинистые покачивания вперед, увеличивая наклон; 4 — вернуться в исходное положение;
- сесть, прямые ноги на ширине плеч, палка на плечах. 1 – повернуть корпус вправо; 2 – поднять палку вверх; 3 – развернуть корпус влево; 4 – опустить палку на плечи. Повторить, затем выполнить упражнение в другую сторону.

***Упражнения с гантелями***

Упражнения с гантелями равномерно воздействуют на все мышечные группы и способствуют их гармоничному развитию (рисунок 2). Необходимо следить за тем, чтобы все движения проделывались правильно, дыхание не задерживалось и напрягались только те мышцы, которые участвуют в данном движении.

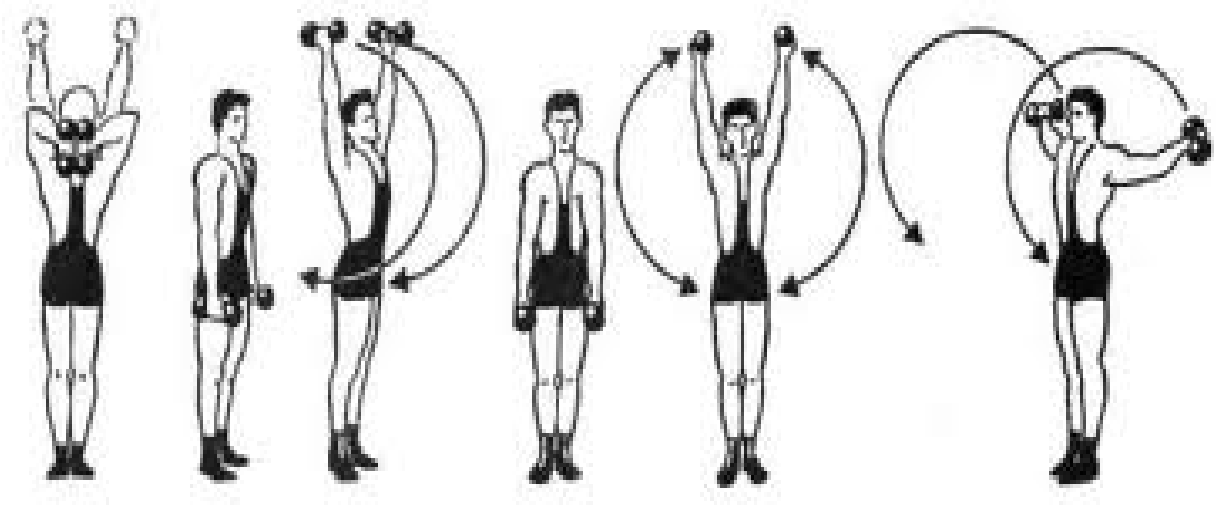


Рисунок 2. Упражнения с гантелями

***Упражнения со скакалкой***

Скакалка – простейшая тренировка, доступная каждому в любом месте и в любое время. Это один из лучших, если не самый лучший, метод повышения уровня физической подготовки, почти не имеющий каких-либо границ или ограничений (рисунок 3).

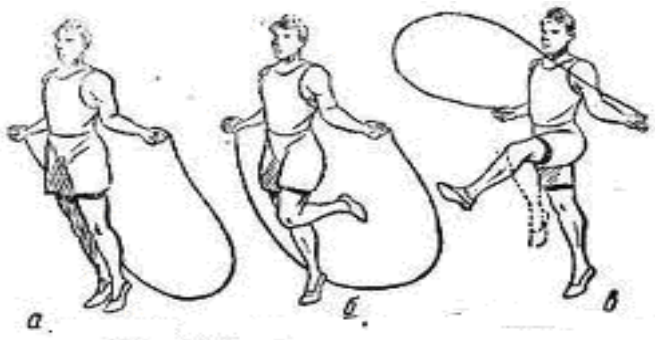



Рисунок 3. Упражнения с гимнастической скакалкой:
а) на двух одновременно; б) на одной ноге; в) бег на месте с различными движениями ног

Упражнения со скакалкой развивают выносливость, укрепляют сердечно-сосудистую и дыхательную системы, развивают прыгучесть, укрепляют мышцы ног, делают фигуру стройной и привлекательной. Можно осуществлять высокий шаг, чередование прыжков с правой ноги на левую, колени поднимаются до пояса, ноги в стороны, чередование прыжков, когда ноги вместе, с прыжками с разведенными в стороны на ширину плеч ногами. Можно выполнять ножницы-прыжки, при которых ноги разводятся не в стороны, а по линии вперед-назад и др. Упражнения со скалкой выполняются самостоятельно.

## 5 РЕКОМЕНДАЦИИ ПО ТЕХНИКЕ БЕЗОПАСНОСТИ НА ЗАНЯТИЯХ ПО ФИЗИЧЕСКОЙ КУЛЬТУРЕ И СПОРТУ

Тесты по физической культуре и спорту необходимо проводить по команде преподавателя при высоком организационном уровне и дисциплине обучаемых, полностью исключающих возможность получения травм.

Основными мерами предупреждения травм являются следующие.

1. Соблюдение установленных правил организации и проведения занятий.
2. Правильная методика обучения.
3. Оказание помощи и поддержка.
4. Страховка и самостраховка.
5. Хорошее материальное обеспечение занятий.
6. Систематический врачебный контроль и самоконтроль.
7. Строгое соблюдение установленной формы одежды.
8. Недопущение нарушения дисциплины, выполнение упражнений только по указанию педагога.

Учитывается физическая подготовка обучающихся, соблюдается постепенность в повышении физической нагрузки.

## 6 ПРИМЕРНЫЙ ПЕРЕЧЕНЬ ТЕМ РЕФЕРАТА

1. Система физической культуры и спорта в Российской Федерации:
2. Физическая культура в профессиональной подготовке обучающихся.
3. Основы законодательства в физической культуре и спорте.
4. Международные нормативные правовые акты по физической культуре и спорту.
5. Нормативные правовые акты по физической культуре и спорту Российской Федерации.
6. Нормативные правовые акты по физической культуре и спорту субъектов Российской Федерации.
7. Медико-биологические основы физического воспитания и здоровый образ жизни.
8. Общие требования врачебного контроля над здоровьем обучающихся.
9. Гигиенические требования и средства восстановления.
10. Основы медицинского контроля и самоконтроля.
11. Первая помощь при травмах.
12. Методы закаливания и поддержания здорового образа жизни.
13. Психологические особенности студентов в процессе занятий физической культурой и спортом.
14. Формирование умственных, морально-волевых, психологических качеств на занятиях по физической культуре и спорту.
15. Финансовое обеспечение физической культуры и спорта.
16. Спортивный маркетинг, спортивное спонсорство, спортивное лицензирование.
17. Деятельность Международного олимпийского комитета, международных спортивных организаций, объединений, ассоциаций и союзов.
18. Деятельность Международной федерации университетского спорта (ФИСУ).
19. Деятельность Олимпийского комитета России.
20. Принципы, методы и величина нагрузки в процессе тренировки по одному из видов спорта.
21. Оригинальные методики развития и совершенствования физических и специальных качеств.
22. Профилактика предупреждения травматизма в процессе спортивных тренировок и соревнований по отдельным видам спорта.
23. Организация и методика проведения физкультурно-спортивных мероприятий в образовательной организации высшего образования.

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**
**ПО ДИСЦИПЛИНЕ «ФИЗИЧЕСКАЯ КУЛЬТУРА И СПОРТ»**

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.П. Уварова
Оператор компьютерной верстки Е.Д. Кожевникова

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ПОРЯДОК ВЫПОЛНЕНИЯ, ОФОРМЛЕНИЯ И ЗАЩИТЫ

**МОСКВА 2014**

Разработано М.А. Лямзиным, д.п.н., проф.;
М.В. Вольфман, к.п.н.;
В.Г. Ерыковой, к.п.н.

Рекомендовано Учебно-методическим
советом в качестве методических указаний
для обучающихся и преподавателей

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ПОРЯДОК ВЫПОЛНЕНИЯ, ОФОРМЛЕНИЯ И ЗАЩИТЫ

Целью методических указаний является предоставление всем участникам образовательного процесса необходимой методической помощи по выполнению, оформлению и защите выпускной квалификационной работы (ВКР).

В методических указаниях сформулированы основные требования к ВКР бакалавра и специалиста, определены цели, задачи и формы выполнения ВКР; приведены рекомендации по выбору темы работы, этапам ее выполнения, объему, структуре, оформлению, а также процедуре защиты.

**О Г Л А В Л Е Н И Е**

## 1 ОБЩИЕ ПОЛОЖЕНИЯ

**Выпускная квалификационная работа** (ВКР) – завершенная научно-квалификационная учебно-исследовательская работа выпускника вуза по определенной теме (проблеме), направленная на систематизацию, закрепление и расширение у него знаний, формирование и развитие навыков и умений самостоятельного решения конкретных научных задач, характеризующая итоговый уровень квалификации и подтверждающая готовность к профессиональной деятельности.

Итоговая аттестация выпускников осуществляется государственными аттестационными комиссиями, формируемыми базовым вузом в установленном порядке. Защита ВКР проводится на открытом заседании экзаменационной комиссии с участием не менее двух третей ее состава в базовом вузе или в одном из региональных центров аттестации образовательной организации.

На основании Положения об итоговой государственной аттестации выпускников высших учебных заведений Российской Федерации, утвержденного приказом Минобразования России от 25 марта 2003 г. № 1155, выпускные квалификационные работы выполняются в формах, соответствующих определенным ступеням высшего образования :

- для квалификации (степени) «бакалавр» – в форме бакалаврской работы;
- для квалификации «специалист» – в форме дипломной работы (проекта).

Бакалаврские работы могут основываться на обобщении выполненных курсовых работ и проектов и подготавливаться к защите в завершающий период теоретического обучения.
Выпускные квалификационные работы, выполненные по завершении основных образовательных программ подготовки специалистов подлежат рецензированию. Порядок рецензирования устанавливается образовательной организацией. На выполнение, оформление и защиту ВКР отводится время, установленное в соответствии с учебными планами направлений подготовки бакалавров и специалистов.

При выполнении ВКР обучающемуся необходимо помнить, что он лично отвечает за качество её подготовки и оформление.

Выполнение ВКР решает следующие задачи:

- развитие познавательных, исследовательских, организаторских и коммуникативных способностей;
- закрепление, расширение, систематизация и интеграция у них теоретических и практических знаний, развитие навыков их применения при решении различных задач в избранном направлении подготовки;
- развитие навыков самостоятельной научной работы и овладение методикой проведения исследований при решении профессиональных проблем;
- оценивание уровня подготовленности выпускников к профессиональной деятельности;
- презентация навыков и умений публичной дискуссии, защиты научных идей, теоретических выводов, практических предложений и рекомендаций.

Представляемая к защите ВКР бакалавра/специалиста должна соответствовать области, объектам, видам и задачам его профессиональной деятельности, определенных соответствующим ФГОС/ГОС.

## 2 ОРГАНИЗАЦИЯ РУКОВОДСТВА И КОНСУЛЬТАТИВНОЙ ПОМОЩИ ПРИ ВЫПОЛНЕНИИ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

### 2.1 Руководство ВКР

В соответствии с Положением об итоговой государственной аттестации выпускников высших учебных заведений Российской Федерации, для подготовки ВКР обучающемуся назначается руководитель, который утверждается приказом ректора образовательной организации. Руководителем могут назначаться профессора, доценты, старшие преподаватели или преподаватели базового вуза и центра доступа. К руководству могут привлекаться практические работники из областей деятельности, к которым ведется подготовка выпускника, имеющие ученую степень, или высококвалифицированные специалисты с большим опытом работы в

соответствующей области деятельности, но не имеющие ученой степени.

Руководитель ВКР:

– оказывает помощь обучающемуся в выборе темы ВКР, формулировке объекта и предмета, цели и задач, гипотезы и других элементов введения ВКР, а также при составлении библиографии по теме;

– оценивает и корректирует (в случае необходимости) предложенный обучающимся проект плана работы над ВКР, разбивки ВКР на формулировки разделов и подразделов, определяет их примерные объемы, сроки представления в первом варианте;

– рекомендует список научной литературы, нормативных правовых актов и других источников по теме ВКР для изучения и использования при выполнении ВКР; помогает выделить наиболее важные из них; ориентирует обучающегося на составление полной библиографии по теме, изучение практики и т.д.;

– проводит консультации, на которых обсуждает с обучающимся результаты проделанной работы, возникшие трудности и проблемы, дает рекомендации по их преодолению;

– определяет готовность ВКР к защите и представляет на неё отзыв.

**2.2 Организация консультаций**

Образовательный процесс в образовательной организации реализуется с помощью электронного обучения и дистанционных образовательных технологий с использованием информационно-телекоммуникационных и Ровеб-технологий. Это дает возможность проводить консультации руководителей ВКР посредством системы IP-хелпинга – индивидуального асинхронного взаимодействия педагогов с обучающимися через Интернет, во время которого обучающиеся задают вопросы руководителю ВКР (преподавателю учебной дисциплины), а руководитель (преподаватель) размещают ответы на специальном сайте в течение 3-4 дней.

Консультации в системе IP-хелпинг, как правило, посвящаются решению таких задач, как:

– формированию структуры ВКР (соответствие наименований разделов и подразделов выбранной теме, разработанному обучающимся рабочему плану);

– оказанию помощи в составлении списка литературы;

– определение правильности формулировок объекта и предмета, целей и задач, гипотезы и методов исследования, содержания приложений и т.п.

Консультации в системе IP–хелпинг доступны обучающимся на сайте «Личная студия» (https://edu.muh.ru/) в разделе «Обучение».

Порядок работы в системе IP-Хелпинг регулируется технологическим документом, действующим в образовательной организации («Открытая Автоматизированная информационная система (ОАЗИС). «Система IP-Хелпинг». Руководство пользователя»)[*].

Консультирование возможно посредством телетьюторингов путем использования слайд-тьюторингов – учебного и методического материала в виде слайд-лекций, обеспечивающих подготовку обучающихся к выполнению научно-исследовательских работ, сдаче экзаменов и выполнению ВКР, а также других видов учебных занятий по интересующей их проблеме. Это дает возможность в индивидуальном режиме активно вести поиск ответов на возникающие вопросы по выбору темы, поиску литературы, анализу современного состояния научных и практических достижений в области выбранного направления исследования и др.

Консультирование также возможно при помощи электронной почты в сети Интернет и может осуществляться лично, при непосредственном контакте обучающегося с руководителем ВКР.

Взаимодействие руководителя с выпускниками можно строиться нижеследующим образом:

Этап 1. Согласование плана (оглавления) работы в соответствии с утвержденной темой.

Срок: первая-вторая неделя срока итоговой аттестации (в зависимости от направления обучения)

Этап 2. Согласование проекта содержания глав ВКР.

[*] Прежде чем задать вопрос, обучающийся должен просмотреть перечень вопросов, сформулированных ранее другими обучающимися, и найти искомый ответ в системе «ОАЗИС» или в телекоммуникационной двухуровневой библиотеке (Фонд «IP-Хелпинг»).

Срок: от третьей–четвертой до третьей-восьмой недели срока итоговой аттестации

Этап 3. Согласование проекта «Введения», «Заключения», «Списка использованной литературы» («Библиографии»), «Демонстрационного материала».

Срок: за две недели до защиты ВКР.

Этап 4. Представление доработанной ВКР и всех необходимых для защиты ВКР бланков (Титульный лист, задание, отзыв, титульный лист к демонстрационному материалу, заполненных в части «ФИО и темы ВКР).

В Задании должен быть отражен утвержденный ранее руководителем календарный план выполнения работы для заполнения и подписи руководителем.

Срок: не позднее, чем за неделю до предзащиты ВКР.

По мере прохождения календарного плана обучающийся обязан исправить все замечания руководителя ВКР и его предложения по доработке ВКР и представить на рассмотрение окончательный, исправленный,доработанный вариант ВКР для получения отзыва и подписания всех не необходимых бланков к ВКР.

Различные виды консультирования (в системе IP-хелпинг, посредством телетьюторингов, при помощи электронной почты и личном контакте) позволяют обучающемуся и руководителю ВКР оперативно связываться друг с другом. Этим обеспечивается требуемое качество выполнения ВКР в любом центре доступа, не зависимо от местонахождения обучающегося.

## 3 ТРЕБОВАНИЯ К ВЫПОЛНЕНИЮ И СОДЕРЖАНИЮ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

### 3.1 Этапы выполнения ВКР

Процесс выполнения ВКР включает в себя ряд взаимосвязанных этапов:

- выбор обучающимся темы ВКР (Приложение А);
- утверждение приказом ректора образовательной организации обучающемуся темы ВКР, назначение руководителя ВКР;
- формирование обучающимся структуры и календарного графика выполнения работы, согласование с руководителем ВКР;
- сбор, анализ и обобщение обучающимся необходимых материалов по выбранной теме ВКР;
- формулирование предварительных теоретических выводов, практических рекомендаций по результатам анализа;
- подготовка первого варианта ВКР и представление его руководителю;
- доработка первого варианта ВКР с учетом замечаний руководителя;
- чистовое оформление ВКР, списка использованных документальных источников и литературы, глоссария и приложений;
- предоставление ВКР на рецензию (только для специалитета).
- подготовка доклада для защиты ВКР на заседании аттестационной комиссии;
- подготовка демонстрационного / раздаточного материала (образец титульного листа – форма ДМ-01), включающего в сброшюрованном виде распечатки схем, графиков, диаграмм, таблиц, рисунков и т.п. на листах формата А4;
- прохождение предзащиты ВКР в виде учебных занятий «Электронная письменная предзащита», «Электронная устная предзащита», «Предзащита выпускной квалификационной работы».

### 3.2 Выбор темы ВКР

ВКР является заключительным этапом подготовки бакалавров/специалистов в вузе. Выполнение и защита ВКР подтверждает готовность выпускника самостоятельно решать профессиональные задачи. В этой связи важная роль принадлежит правильному выбору темы ВКР.

Тематика ВКР определяется централизованно базовым вузом. Выбор темы ВКР осуществляется выпускником исходя из уровня понимания и осознания её актуальности, оценки теоретического и практического значения. Обучающийся может выбрать тему из предлагаемого перечня тем ВКР или может предложить свою тему исходя из собственных научных и практических интересов, не выходя за рамки направления подготовки и обосновав руководителю ВКР целесообразность её разработки. Наименование инициативной темы ВКР должно быть в установленном порядке утверждено базовым вузом.

В любом случае тема ВКР должна быть актуальной, иметь теоретическую значимость и практическую направленность, соответствовать современному состоянию и перспективам развития науки, техники и культуры.

Свобода выбора тем ВКР позволяет реализовать индивидуальные научные и практические интересы выпускника.

При выборе темы обучающийся должен:

- убедиться в доступности всех необходимых материалов по выбранной теме ВКР;
- выявить наличие не изученных или мало изученных проблем теоретического и практического характера в избранной области исследования;
- определить уровень собственной подготовленности по теме ВКР при выборе вида исследования: теоретического или практического.

### 3.3 Структура и объем ВКР. Разработка рабочего плана

ВКР является самостоятельной учебно-исследовательской работой обучающегося и должна характеризоваться выполнением следующих требований:

– четкой целевой направленностью;

– логической последовательностью изложения материала;

– краткостью и точностью формулировок;

– конкретностью изложения результатов исследования;

– доказательностью теоретических выводов и обоснованностью практических рекомендаций;

– грамотным изложением и оформлением текста ВКР.

Для составления рабочего плана написания ВКР обучающийся должен хорошо представлять ее структуру, которая имеет следующий вид:

- содержание;
- введение;
- основная часть;
- заключение;
- глоссарий;
- список сокращений;
- список литературы;
- приложения.

Объем ВКР (без приложений) должен составлять для бакалавров 50-70 страниц, для специалистов – 90-110 страниц выровненного «по ширине» компьютерного текста. (Приложение Б. Унифицированные требования к

оформлению выпускных квалификационных работ). Написание научно-исследовательских работ регламентируется соответствующими методическими указаниями (1498.01.01;МУ.01;5 «Введение в технологию обучения образовательной организации. Основные правила оформления учебно-научных и творческих работ»).

ВКР, как любое научное исследование, предполагает наличие плана ее осуществления. Выполнение обучающимся ВКР начинается с составления им рабочего плана, представляющего своеобразную наглядную схему (порядок, последовательность, алгоритм) предпринимаемого исследования.

Правильно составленный рабочий план позволяет продуктивно организовать исследовательскую работу по избранной теме ВКР и представить ее к защите в установленные сроки. Рабочий план выполнения ВКР составляется параллельно с отбором и анализом научной литературы. Он согласовывается с руководителем ВКР и имеет произвольную форму, позволяющую включать в него новые аспекты, появляющиеся в процессе выполнения ВКР.

### 3.4 Информационный и библиографический поиск, сбор, анализ и обобщение публикаций

Работа по выполнению ВКР начинается с формирования концептуального авторского замысла научного исследования, который отражается в рабочем плане обучающегося и оглавлении ВКР, а также со сбора и изучения публикаций – документов, доступных для массового использования. К публикациям относятся различные источники и научная литература. Сбор источниковой базы и научной литературы по теме ВКР должен сопровождаться формированием библиографического списка.

Источниками для формирования библиографического списка могут быть:

- перечень рекомендованной в качестве обязательной и дополнительной литературы по теме ВКР;
- электронные образовательные ресурсы в сети Internet;
- библиографические списки и сноски в учебниках, учебных пособиях, диссертациях, монографиях, научных статьях и т.п. по тематике ВКР;
- источники, рекомендованные руководителем ВКР;
- каталоги телекоммуникационной двухуровневой библиотеки (ТКДБ) образовательной организации и библиотек, к которым ТКДБ предоставляет доступ в режиме виртуального читального зала.

В первую очередь следует подбирать литературу и источники за последние 5 лет для гуманитарной и 7 лет для естественно-научной и технической тематики, поскольку в них, как правило, отражены последние научные достижения по проблеме (теме) исследования, представлено современное законодательство и обобщен опыт практической деятельности. Использование литературных и иных источников, изданных в более ранние периоды времени, должно быть скорректировано применительно к современным концепциям ученых и специалистов, реалиям современной жизни.

Указание на научные источники по исследуемой теме можно обнаружить в сносках и в списке литературы уже изданных работ. Поиск статей в научных журналах следует осуществлять путем просмотра последнего номера соответствующего журнала за определенный год, так как в нем, как правило, помещается указатель всех статей, опубликованных в данном журнале за прошедший год. При выполнении ВКР особенно внимательно следует изучать профессиональные и специализированные периодические издания (журналы, газеты, сборники научных трудов).

При выполнении ВКР обучающийся имеет возможность работать с литературой по теме, используя ТКДБ образовательной организации, потому что доступ к ее ресурсам возможен с сайта «Личная студия», а также из любого центра доступа. ТКДБ предоставляет возможность в режиме виртуального читального зала изучать ресурсы удаленного доступа таких электронных библиотек, как:

- Научная электронная библиотека (НЭБ);
- Открытая русская электронная библиотека;
- Единое окно доступа к образовательным ресурсам;

– Электронная библиотека международных документов по правам человека;
– База электронных диссертаций "Proquest digital dissertations";
– Портал «Theses Canada» («Канадские полнотекстовые диссертации»);
– База журналов открытого доступа «Directory of open access journals» и др.

Для выполнения ВКР большой интерес представляет «Единое окно доступа к образовательным ресурсам». В этой электронной библиотеке размещены образовательные информационные ресурсы, разработанные ведущими российскими вузами: учебники, учебные пособия, тексты лекций, методические рекомендации, указания и др.

В ТКДБ представлен широкий круг научных журналов на русском языке по всем областям знаний. Пользователь имеет доступ к алфавитному перечню заглавий журналов и возможность отбора по году выпуска журнала. Фонд справочных, нормативных и официальных изданий ТКДБ содержит отраслевые и универсальные энциклопедии, словари, справочники.

В образовательной организации оформлена подписка на коллекцию журналов «Научной электронной библиотеке» по социальным и гуманитарным наукам. Коллекция содержит журналы по социологии, психологии, юриспруденции, образованию, менеджменту и др. Преодолеть языковой барьер поможет система компьютерного перевода в Google.

Работа с научной книгой начинается с изучения титульного листа, где приводятся данные об авторе и выходные сведения (год и место издания), а также с аннотации и оглавления. Год издания книги позволяет соотнести информацию, содержащуюся в ней, с существующими знаниями по данной проблеме на настоящее время. В аннотации и оглавлении книги раскрываются ключевые моменты ее содержания, логика и особенности изложения материала.

Далее необходимо ознакомиться с введением книги, где, как правило, формулируется актуальность темы, кратко излагается содержание и направленность, раскрываются источники и способы исследования, другие атрибуты научного познания.

Ознакомление можно завершить постраничным просмотром, обратив внимание на научный аппарат, частично расположенный в сносках, на определения ключевых понятий, полноту изложения заявленных в оглавлении проблем.

При изучении специальной научной литературы необходимо обращаться к энциклопедиям, словарям и справочникам в целях выяснения смысла специфических терминов и понятий, выписывая (конспектируя) те из них, которые в дальнейшем будут использованы в тексте ВКР и при составлении глоссария.

Изучение и использование при выполнении ВКР нормативных документов – законов, подзаконных актов, постановлений – является обязательным, так как знание этих документов и умение работать с ними – залог успешной профессиональной деятельности выпускника.

В виртуальном читальном зале ТКДБ обучающимся предоставляется возможность удаленного доступа к информационным ресурсам «Электронной библиотеки международных документов по правам человека», в которой размещается информация о различных межправительственных организациях в области прав человека, о проводимых и планируемых конференциях, сессиях органов по контролю за соблюдением международных договоров в области прав человека, а также оперативная информация о принятых решениях, рассматриваемых докладах и отчетах о соблюдении прав человека.

Образовательная организация, являясь пользователем справочно-информационных системы «Консультант Плюс» или «Гарант», предоставляет возможность каждому обучающемуся быть в курсе последних изменений в законодательстве и решать возможные проблемы в области правовой информации и бухгалтерской документации. Данные системы являются самыми обширными правовыми базами России, которые содержат не только нормативные правовые акты, составляющие основу российского законодательства, но и уникальный банк консультаций экспертов в области налогообложения, обзоры судебной и арбитражной практики, деловую документацию.

В ходе анализа собранного по теме ВКР материала обучающиеся делают обоснованные и аргументированные конспективные записи, выписки, цитаты и систематизируют их по ключевым вопросам

исследования. На основе обобщенных данных они уточняют структуру, содержание и объем ВКР, информируют руководителя ВКР о планируемых коррективах в работе.

### 3.5 Характеристика структурных частей ВКР

Каждая структурная часть ВКР (содержание, введение, основная часть, заключение, глоссарий, список сокращений, список использованных источников, приложения) имеет свое назначение. Оформляя ВКР, автор должен помнить, что каждая структурная часть начинается с новой страницы.

**Содержание** *ВКР включает заголовки всех разделов (глав, параграфов и т.д.), содержащихся в ВКР. Обязательное требование: дословное повторение в заголовках содержания названий разделов, представленных в тексте, и наоборот, в той же последовательности и соподчиненности. Примеры содержаний для выпускных квалификационных работ бакалавра юриспруденции (Приложение В) и дипломированного специалиста менеджмента (Приложение Г) приводятся в конце методических указаниях.*

Во ***введении*** ВКР обосновывается *актуальность* выбранной темы. Обосновать актуальность – значит аргументированно объяснить и доходчиво доказать, что выбранную обучающимся тему ВКР необходимо и важно изучать в настоящее время как с теоретической, так и с практической точек зрения. Обоснование актуальности темы требует от автора ВКР ответов на следующие вопросы: Что определило выбор темы? Чем эта тема интересна для обучающегося в данный момент времени? Почему её изучение и выполнение по ней ВКР является своевременным и необходимым? Какое приращение теоретического знания даст проведение данного исследования? Какое значение для улучшения практики имеет выполнение ВКР?

Во введении также формулируется и кратко характеризуется *основное (ведущее) противоречие* в той сфере теории и практики, которой посвящена тема ВКР. Основное (ведущее) противоречие – главное несоответствие, несовпадение между тем, что и как должно быть («между должным»), и тем, что и как существует на самом деле («между сущим»). Основное (ведущее) противоречие составляет суть *научной проблемы*, а её решение – смысл *научной задачи* ВКР.

Далее во введении представляется *степень разработанности темы (научной проблемы) ВКР*. Дается краткий обзор источников и научной литературы. Анализируется степень разработанности выбранной темы исследования в целом или отдельных аспектов в проведенных научных исследованиях. Выявляется её недостаточная изученность на современном этапе развития общества и на возможную перспективу. Показывается необходимость изучения научной проблемы в новых социально-экономических, политических, культурных, образовательных и иных условиях. В результате анализа степени разработанности темы (научной проблемы) автор должен сделать логический вывод о том, что именно они недостаточно раскрыты в теории и изучены на практике, что и требует дальнейшего исследования в рамках ВКР.

Кроме этого, во введении ВКР формулируются объект и предмет, цель и задачи, гипотеза исследования, указываются избранные автором методы познания, определяется практическая значимость полученных результатов.

*Объект исследования* – это явление (процесс, деятельность, система), которое автор ВКР избрал для изучения. Объект исследования отвечает на вопрос: «*Что* рассматривается?». При этом следует иметь в виду, что один и тот же объект исследования может изучаться многими исследователями. Однако новизна, оригинальность и значимость каждого исследования характеризуется предметом исследования.

*Предмет исследования* – это аспект, грань, сторона, часть изучаемого явления – объекта, на которую непосредственно направлено внимание исследователя. Как правило, предмет исследования и тема ВКР по своему смыслу совпадают.

Объект исследования шире, чем его предмет; предмет исследования находится в границах объекта; рамки предмета исследования не должны «выходить» за объект.

Для изучения объекта и предмета исследования формулируются цель и задачи ВКР.

*Цель* исследования – это мысленно предвосхищаемый (прогнозируемый) автором целостный образ конечного результата; это предполагаемый итог всей проделанной работы, от её начала до конца. Цель исследования, особенно ВКР бакалавров, должка быть сформулирована таким образом, чтобы полученные результаты удовлетворяли практические потребности людей в решении актуальной научной задачи (темы ВКР). Формулировка цели исследования обычно начинается словами «обосновать…», «разработать…», «выявить…» и далее: особенности, условия, факторы, методику, модель, методы, механизмы, критерии, требования, технологию и т.п. При этом цель исследования должна коррелировать с названием темы ВКР и предметом исследования.

*Задачи* исследования – это прогнозируемый автором образ промежуточных результатов; это предполагаемый итог конкретной части (этапа, периода) работы исследователя. Задачи исследования определяются поставленной целью, они находятся в целевом поле исследования и их конкретизируют. Решение задач исследования, в конечном счете, позволяет добиться цели исследования. Формулировка задач исследования обычно начинается словами: проанализировать подходы к …, обобщить точки зрения на …, систематизировать имеющиеся позиции по …, разработать классификацию …, установить зависимости …, выявить состояние …, разработать предложения … и т.п. При этом задачи исследования должны коррелировать с названиями разделов и подразделов оглавления ВКР.

*Гипотеза* исследования – предположение, выдвигаемое для объяснения того, как можно преобразовать (изменить, совершенствовать, улучшить) изучаемое явление (предмет исследования); это представление обобщенных теоретических положений, основных идей и результатов, к которым может привести исследование. Гипотеза формулируется после того, как автор изучил источники и научную литературу по теме ВКР, практику функционирования исследуемого явления и выявил ведущее (основное) противоречие. Формулировка гипотезы исследования обычно включает такую теоретическую конструкцию: предполагается, что разрешить выявленное противоречие возможно, если …

Далее во введении представляются *методы исследования* – это способы познания, позволяющие достигнуть цель, решить задачи и доказать гипотезу исследования; это своеобразные инструменты и механизмы нахождения и накопления фактического (эмпирического) материала, его анализа и объяснения, обоснования условий, факторов, путей, направлений и т.п. преобразования изучаемого явления.

Основными методами научного исследования являются:

- анализ источников и научной литературы;
- обобщение отечественной и зарубежной практики;
- систематизация различных теорий, концепций, подходов;
- моделирование изучаемого явления;
- сравнение (компаративистский метод);
- наблюдение и его разновидности (индивидуальное и групповое, кратковременное и длительное, непосредственное и опосредованное, включенное и др.);
- опросные методы (интервьюирование, анкетирование, тестирование и т.д.);
- анализ результатов (продуктов) деятельности;
- экспериментальные методы.

Автору ВКР целесообразно перечислить только те методы исследования, которые действительно нашли применение в данной работе.

Формулировка *практической значимости* ВКР должна свидетельствовать о том, каким образом, где и кем можно использовать полученные в исследовании конкретные результаты в практической деятельности.

Введение завешается представлением структуры ВКР. Автор пишет: ВКР состоит из введения, двух (или трех) глав, заключения и т.д.

Объем введения для ВКР бакалавра составляет 2-4 стр., для специалиста – 5-7 стр.

Необходимо отметить важную рекомендацию: окончательное оформление введения целесообразно делать после завершения выполнения основной части и заключения ВКР.

***Основная часть*** ВКР должна соотноситься с поставленными целью и задачами. В зависимости от того, какие задачи стоят перед автором, основная часть делится на 2 или 3 главы. Объем глав основной части должны быть соразмерны друг другу. Деление глав на параграфы необязательно, но возможно, если в этом есть необходимость.

Предварительная структура основной части ВКР (главы, параграфы) определяется еще на стадии планирования работы. Однако в ходе выполнения ВКР могут возникнуть новые идеи и соображения, которые потребуют не только изменить и уточнить структуру, но и обогатить содержание ВКР и увеличить ее объем.

Содержанием основной части исследования является теоретическое осмысление научной проблемы и изложение фактического эмпирического материала. Последовательность изложения того и другого может быть различной. Все зависит от авторской концепции исследования (плана работы), согласованной с руководителем ВКР. Чаще вначале излагаются основные теоретические положения по исследуемой теме (этому посвящается первая глава ВКР), а затем – эмпирический материал, результаты экспериментальной работы и т.п., которые подтверждают изложенную теорию. Но возможна и другая последовательность изложения, когда вначале анализируется фактический материал, а затем делаются теоретические обобщения и выводы.

Как правило, в отдельный параграф основной части исследования выделяется анализ публикаций по теме исследования: источников и научной литературы. Источники – это тексты, которые являются специальным предметом исследования: исторические (архивные, мемуарные) документы, законодательные и иные нормативные акты. Научная литература – это публикации, которые используются при выполнении ВКР, но при этом не являются предметом исследования: учебники, учебные пособия, диссертации, монографии, статьи и т.п. Те и другие могут быть как в печатном, так и в электронном (цифровом) виде. Умение различать эти две группы публикаций чрезвычайно важно.

Излагать материал следует своими словами, но грамотным русским языком. Допускается умеренное цитирование различных публикаций с обязательными ссылками на автора(ов) и сами публикации. **Недобросовестное заимствование текстов и результатов исследований у других авторов не допускается**! Сноски в тексте печатаются одинарным межстрочным интервалом, размер (кегль) - 12.

Связь между абзацами в основной части ВКР обеспечивается как общей логикой рассмотрения темы (научной проблемы), так и специальными выражениями-связками, например:

- Анализ научных источников свидетельствует, что ...
- Исследование практической деятельности показало ...
- Важное значение в рамках современных подходов к ...
- Важнейшим элементом рассматриваемого явления (процессов, системы и т.п.) является...
- Подобные исследования, проведенные в рамках ...
- Анализ научной литературы, изучение практики реализации ...
- Специалисты по этой проблеме сделали вывод о том, что ...
- В связи с тем, что ...
- Изучение данного вопроса дает возможность утверждать ...

В конце каждой главы должны быть сформулированы краткие выводы как результаты исследования, которые, как правило, начинаются словами «Таким образом, ...», «Итак, …», «Следовательно, …». Выводы по главам ВКР должны коррелировать с задачами исследования.

Объем основной части ВКР бакалавров – 40-50, специалистов – 80-100 страниц компьютерного текста.

После основной части ВКР пишется ***заключение,*** которое обусловлено логикой проведения исследования, носит форму обобщения и синтеза накопленной в основной части теоретической и практической информации.

Заключение должно содержать краткую формулировку результатов, полученных в ходе исследовательской работы. Поэтому основные положения заключения ВКР должны коррелировать с целью и задачами исследования. Текст заключения не должен дублировать выводы по главам. В нем на новом, более высоком уровне обобщения представляются теоретические выводы и практические рекомендации, которые вытекают из проведенного исследования.

Объем заключения примерно равен объему введения.

***Глоссарий.*** При выполнении ВКР предусмотрено составление глоссария, являющегося её обязательным компонентом. Для бакалаврской работы он должен содержать 15-20 основных понятий и терминов, используемых в контексте исследуемой проблемы, для дипломной работы – не менее 25.

В глоссарий включаются основные профессиональные термины (а также их английские или латинские аналоги, в необходимых случаях – аналоги на других языках), персоналии, важнейшие даты истории и т.п. При подготовке глоссария авторы могут использовать энциклопедии, словари, справочники, документы законодательного характера и др. Используя в тексте ВКР термины, уместно применяя и правильно раскрывая их содержание, автор демонстрирует свою профессиональную компетентность.

***Список использованных источников*** является обязательным атрибутом ВКР и отражает уровень самостоятельной творческой деятельности обучающегося.

В этот раздел в обязательном порядке включается библиографическое описание всех цитированных или упоминаемых в тексте ВКР публикаций (законодательных документов и нормативных актов, монографий и другой научной литературы). В качестве исключения могут быть включены публикации, которые были изучены автором при выполнении ВКР и которые оказали влияние на выработку авторской концепции, но о них нет упоминания в тексте.

В списке использованных источников ВКР бакалавров следует привести примерно 30-50 наименований публикаций.

***Список использованных источников ВКР специалистов должен содержать не менее 50 наименований публикаций.***

***Список сокращений*** составляется при необходимости и включает расшифровку наиболее часто упоминаемых в тексте ВКР сокращенных наименований организаций, документов, понятий, слов и т.д. Например:

***ВОЗ*** – Всемирная организация здравоохранения;

***ЕБРР*** – Европейский банк реконструкции и развития;

***ФГОС*** – федеральный государственный образовательный стандарт.

***Приложения*** являются обязательным компонентом ВКР. В приложениях следует приводить различные вспомогательные материалы (таблицы, схемы, графики, диаграммы, иллюстрации, копии постановлений, договоров, инструкции, вспомогательные расчеты и т.п.). С одной стороны, они призваны дополнять и иллюстрировать основной текст, с другой – разгружать его от второстепенной информации. Все материалы, помещенные в приложениях, должны быть связаны с основным текстом, в котором обязательно делаются ссылки на соответствующие приложения. Например, см. Приложение А.

Каждое приложение начинается с новой страницы и должно иметь надпись **ПРИЛОЖЕНИЕ** и заголовок (название). Образцы оформления приложения см. в конце данного пособия.

Количество страниц приложений не входит в требуемый объем ВКР. Страницы приложений не нумеруются.

### 3.6 Требования к оформлению ВКР

Этап оформления ВКР является не менее важным, чем остальные, так как на этом этапе автор должен не только свести все материалы в единый документ, но и оформить их в соответствии с требованиями. Правила, регламентирующие оформление учебно-научных и творческих работ, а также оформление научно-справочного аппарата к ним (цитаты, ссылки, сноски, список источников и научной литературы), обязательные для соблюдения обучающимися образовательной организации, изложены в методических рекомендациях (1498.01.01;МУ.01;5 «Введение в технологию обучения. Основные правила оформления учебно-научных и творческих работ»).

К оформлению окончательного («чистового») варианта ВКР автор приступает тогда, когда все материалы собраны и сделаны необходимые обобщения, а также получено одобрение руководителя ВКР. Далее проверяются и критически оцениваются каждый вывод, формула, таблица, каждое предложение и каждое

отдельное слово. Необходимо еще раз тщательно проверить и отредактировать текст, устранить выявленные ошибки, описки, опечатки. Далее следует проверить логику работы – насколько точен смысл абзацев и отдельных предложений, соответствует ли содержание глав, параграфов их заголовкам.

Затем следует проверить, нет ли в работе пробелов в изложении и аргументации, устранить стилистические погрешности, обязательно проверить точность цитат и ссылок, правильность оформления, обратить внимание на написание числительных и т.д. Лишь после такой корректуры следует подготовить окончательный вариант ВКР. Тщательная и грамотная отработка текста ВКР свидетельствуют об ответственности автора за представляемый материал, его уважении к руководителю и членам аттестационной комиссии, оценивающим работу.

Окончательный вариант ВКР проверяется на учебном занятии вида «Электронная письменная предзащита», в которое входит процедура нормоконтоля, проводимая при помощи ПО «Нормоконтроль» с целью обеспечения единообразия в структуре и оформлении ВКР, а также ее проверки на профессионализм и оригинальность интеллектуальным роботом контроля оригинальности и профессионализма (ИР КОП). Для этого обучающийся самостоятельно загружает электронный вариант ВКР в шаблон «Электронная письменная предзащита», расположенный на сайте «Личная студия» (https://edu.muh.ru/.) для проверки. Если программное обеспечение выявило недочеты в оформлении ВКР, то обучающийся должен внести в нее соответствующие правки.

Шаблон «Электронная письменная предзащита» используется для формирования ВКР в электронном виде для прохождения предзащиты, транспортировки в базовый вуз и последующего хранения.

Структурными элементами шаблона «Электронная письменная предзащита» являются:

- основные сведения о работе;
- содержание;
- введение;
- основная часть;
- заключение;
- глоссарий;
- список использованных источников;
- список сокращений;
- приложения.

Каждый структурный элемент электронной письменной предзащиты ВКР должен начинаться с новой страницы.

Все перечисленные структурные элементы являются обязательными, кроме элемента «Список сокращений» и главы 3 раздела «Основная часть».

ВКР, подготовленная к защите и прошедшая электронную письменную предзащиту, сдается руководителю ВКР.

Руководитель анализирует содержание ВКР на соответствие заявленной теме, оценивает уровень разработанности проблемы, степень использования привлекаемых материалов, правильность структурирования материала, достоверность и обоснованность полученных результатов, аргументированность теоретических выводов, грамотность изложения.

Руководитель дает письменное заключение (отзыв) (форма 19-мд, Приложение Д) о степени соответствия ВКР предъявляемым требованиям. Отзыв – это оценка не только качества ВКР выпускника. Это оценка его учебной и исследовательской деятельности над выбранной темой, активности, самостоятельности, системности мышления, уровня знаний и умений поиска и нахождения нужной информации и пр. Руководитель оформляет готовность выпускника к защите своей подписью на титульном листе ВКР (форма 09-д, Приложение Е).

Если ВКР не представлена руководителю в установленный срок, или обучающийся не допущен к защите ВКР, выпускник отчисляется из образовательной организации как не прошедший итогового аттестационного испытания.

Вместе с оформленной и сброшюрованной ВКР (с обязательной собственной подписью и подписью научного руководителя работы) обучающийся представляет на защиту тщательно оформленные демонстрационные плакаты (или сброшюрованный «раздаточный материал», экземпляры которого передаются каждому члену аттестационной комиссии).

Назначение демонстрационных плакатов («раздаточного материала») в том, чтобы акцентировать внимание членов аттестационной комиссии и присутствующих на результатах, полученных обучающимся при выполнении ВКР. Кроме этого, как свидетельствует практика, наличие демонстрационных плакатов («раздаточного материала») помогает выступающему во время защиты более конкретно и связанно изложить содержание своего доклада.

На демонстрационных плакатах (формат А1) и в «раздаточном материале» (формат А4) отображаются схемы, графики, диаграммы, таблицы и другие данные, характеризующие результаты ВКР. Все выносимые обучающимся на защиту демонстрационные плакаты (в уменьшенном виде) и компьютерные распечатки материалов из «раздаточного материала» обязательно должны присутствовать (дублироваться) в соответствующих разделах ВКР.

На защиту ВКР не допускается представление демонстрационных плакатов и «раздаточного материала», не связанных по своему содержанию с текстом доклада, а как бы «оживляющих» и «украшающих» доклад выпускника. Также не допускается представление на защиту демонстрационных плакатов и информации в «раздаточном материале», на которые нет ссылок в докладе.

Как правило, для иллюстрации результатов выполненной ВКР достаточно 4–6 плакатов или примерно такого же числа страниц компьютерных распечаток в «раздаточном материале».

Образец титульного листа «раздаточного материала» приведен в Приложении Ж. В Приложении И дается примерный перечень информации, которую рекомендуется размещать на демонстрационных плакатах или в «раздаточном материале».

Если в процессе защиты ВКР выпускник использует компьютерную презентацию работы, то она исполняет роль демонстрационного материала.

### 3.7 Подготовка к защите ВКР

Подготовка к защите ВКР представляет собой творческую и ответственную работу выпускника. Важно не только написать высококачественную ВКР, но и уметь ее успешно защитить.

Получив положительный отзыв о ВКР от руководителя ВКР, выпускник должен подготовить доклад (при защите выпускной квалификационной работы время на доклад – 7-10 минут) в котором кратко излагаются основные положения и результаты ВКР. Текст выступления должен быть максимально приближен к тексту ВКР. Поэтому его основу составляют положения, сформулированные во введении, в выводах по главам и в заключении, которые воспроизводятся в выступлении практически полностью.

Выступление на защите ВКР отрабатывается обучающимся на учебных занятиях вида «Электронная устная предзащита» и «Защита выпускной квалификационной работы».

Доклад следует начинать с обоснования актуальности избранной темы, описания научной проблемы, формулировки объекта, предмета, цели и задач ВКР. Далее необходимо сказать об используемых при выполнении ВКР методах исследования, а также по главам раскрыть основное содержание ВКР, обращая особое внимание на полученные наиболее важные и интересные результаты, критически оценивая их.

Заключительная часть доклада основывается на заключении ВКР, на перечислении наиболее общих теоретических выводов без повторения частных обобщений, сделанных при характеристике глав основной части. Доклад не должен быть перегружен цифровыми данными, которые, в случае необходимости, приводятся в демонстрационных плакатах и в «раздаточном материале», а в докладе лишь делаются на них ссылки. Рекомендации к структуре доклада при защите ВКР приведены в Приложении К.

Отработка выпускником текста выступления на публичной защите ВКР проводится в рамках учебных занятий «Устная электронная предзащита» и «Предзащита выпускной квалификационной работы» (Методические указания по подготовке и проведению предзащиты ВКР (9058.01.01;МУ.01;1)).

### 3.8 Рекомендации по составлению компьютерной презентации (ВКР с помощью пакета OpenOffice.Org Impress

В широком смысле слова презентация (англ. presentation – «представление») – это выступление, доклад, защита законченного или перспективного проекта, представление на обсуждение рабочего проекта, результатов внедрения и т.п.

Компьютерная презентация (КП) представляет собой электронный документ в виде упорядоченного и связанного набора отдельных кадров (слайдов), выполненных в технологии мультимедиа. Отдельный слайд может содержать текст, рисунки, фотографии, анимацию, видео и звук.

Использование КП позволяет значительно повысить информативность и эффективность доклада при защите ВКР, способствует наглядности и выразительности излагаемого материала.

Подготовка КП к защите – это ответственная, кропотливая и полезная умственная деятельность обучающегося, которая структурирует мысли материал, позволяет выявить «узкие» места ВКР.

КП позволяет наглядно отображать на экране монитора компьютера или настенном экране в концентрированном виде подготовленный выпускником материал для доклада. Поэтому малейшие недочеты становятся видны.

Положительной стороной создания КП является максимальная собранность обучающегося. Работая с мультимедийными презентационными технологиями, он показывает умение представлять итоги своего научного труда с привлечением современных компьютерных средств, выполняет требования, предъявляемые к уровню подготовки бакалавра/специалиста, изложенные в ФГОС ВПО и ГОС ВПО по различным направлениям подготовки (специальностям).

КП позволяет членам аттестационной комиссии одновременно изучать ВКР и воспринимать доклад выпускника на слух и зрительно.

Доклад на защите ВКР целесообразно сопровождать презентацией с использованием 6-8 слайдов. Каждый слайд должен иметь заголовок, количество слов в слайде не должно превышать 40. Размер шрифта (кегль) в слайде от 28 до 36.

Основными принципами при составлении КП являются: лаконичность представляемой информации; ясность суждений и мыслей автора; наглядность излагаемого материала; оптимальное использование возможностей компьютерной программы OpenOffice,Org Impress (вставок, дизайна, анимации и т.п.).

КП необходимо начать с заголовочного слайда, в котором приводятся название темы ВКР и ФИО автора. В последующих слайдах автор представляет основные положения и результаты выполненной ВКР.

При подготовке КП следует использовать дизайн шаблонов (Формат – Применить оформление). Не следует увлекаться яркими шаблонами, информация на слайде должна быть контрастна фону, а фон не должен затенять содержимое слайда. Не следует злоупотреблять эффектами анимации. Оптимальной настройкой эффектов анимации является появление в первую очередь заголовка слайда, а затем – текста по абзацам. При этом, если несколько слайдов имеют одинаковое название, заголовок слайда должен постоянно «оставаться» на экране. Динамическая анимация эффективна тогда, когда в процессе выступления происходит логическая трансформация существующей структуры в новую структуру, предлагаемую вами. Настройка анимации, при которой происходит появление текста по буквам или словам, может вызвать негативную реакцию со стороны членов комиссии, которые одновременно должны слушать выступление, изучать текст ВКР и воспринимать визуальное представление материала исследования.

Для настройки временного режима презентации используется меню «Показ слайдов» → «Режим настройки времени». Предварительно надо определить, сколько минут требуется на каждый слайд. Очень важно не торопиться при докладе и четко произносить слова. Презентация помогает сделать доклад, но она не

должна его заменять. Желательно подготовить к каждому слайду заметки по докладу (Вид – Страницы заметок). Можно распечатать некоторые ключевые слайды в качестве демонстрационного материала.

## 4 ОЦЕНКА КАЧЕСТВА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

### 4.1 Порядок рецензирования ВКР

В соответствии с ФГОС ВО, а также другими нормативными документами Минобрнауки России и нормативными документами, ВКР дипломированных специалистов подлежат обязательному рецензированию.

Рецензирование - процедура рассмотрения и экспертной оценки рецензентом предлагаемой к защите выпускной квалификационной работы с целью независимой оценки ее качества, полноты раскрытия вопросов и практической ценности выполненной работы, выявления ее достоинств и недостатков, соответствия требованиям Федерального государственного образовательного стандарта по направлению подготовки.

В числе рецензентов могут быть работники министерств, ведомств, предприятий (организаций, фирм), педагогические и научно-педагогическое работники других образовательных, научных организаций, специалисты из числа работодателей, работников других организаций, хорошо владеющих вопросами, связанными с тематикой выпускной квалификационной работы. Основное требование для назначения рецензента – наличие у предполагаемого эксперта высшего образования, ученой степени по соответствующему направлению подготовки и опыта практической работы.

Для экспертизы ВКР специалистов рекомендуется привлекать внешних рецензентов.

Рецензирование ВКР преподавателями выпускающих кафедр образовательной организации не допускается. Рецензент должен рассмотреть направленную ему ВКР в установленные сроки, написать рецензию и предоставить ВКР вместе с рецензией в образовательной организации до издания приказа о допуске к защите.

Рецензирование выпускной квалификационной имеет следующие этапы: проверка соответствия темы направлению подготовки, наличие достаточного количества актуальных теоретических источников, примерного плана, степени раскрытия темы, достижения цели и решения поставленных задач, соответствия выводов и практических рекомендаций содержанию проведенного исследования, проверка логичности изложения материала, правильности оформления работы предъявляемым требованиям.

Оценивание ВКР проходит по следующим критериям, которые характеризуют сформированные у обучающегося умения:

- соответствие темы ВКР направлению подготовки;
- соответствие содержания ВКР избранной теме и плану исследования;
- оценка актуальности, правильности формулирования научного аппарата исследования;
- правильное использование основных методов научного исследования;
- оригинальность выводов и практических рекомендаций, их соответствие проведенному исследованию;
- оформление результатов исследований в соответствии с предъявляемыми требованиями;
- степень самостоятельности магистранта, проявленной при выполнении и оформлении работы по всем ее основным элементам.

Рецензент должен сосредоточить внимание на качестве выполненной работы и изложить в рецензии:

- актуальность и практическую значимость ВКР;

- соответствие содержания работы целевой установке, научный уровень, полноту и качество разработки темы;

- следует отметить те разделы работы, которые характеризуют исследовательские способности выпускника, умение прогнозировать динамику, тенденции развития объекта (процесса, задач, проблем, их систем);

- наличие или отсутствие системности, логической взаимосвязи всех частей выпускной квалификационной работы друг с другом, ясности изложения материала.

- практическую реализацию и выбор инструмента для решения поставленной задачи;

- общую характеристику работы с точки зрения ее завершенности и внедрения на практике;

- отмечается оригинальность принятых решений, элементы новизны и их практическое значение: замечания и по работе;

-конкретная оценка выполненной работы и ее соответствие требованиям ФГОС ВПО (сформированности определенных компетенций)

- следует указать те вопросы (если такие присутствуют в работе), которые не получили достаточного освещения в ВКР.

В заключительной части рецензии дается однозначная оценка выпускной квалификационной работы по четырехбалльной системе (отлично, хорошо, удовлетворительно, неудовлетворительно) и высказывается мнение о возможности присвоения обучающемуся квалификации.

С целью унификации рецензий на ВКР специалистов рекомендуется использовать единую форму рецензии (образец рецензии представлен в Приложении Л).

Рецензия оформляется и подписывается рецензентом с указанием его должности, ученой степени и (или) ученого звания (при наличии). Отрицательный отзыв рецензента не является препятствием для защиты выпускной квалификационной работы. Внесение изменений в дипломную работу после рецензирования не допускается. Рецензия зачитывается при защите квалификационной работы, и мнение рецензента учитывается при определении окончательной оценки.

### 4.2 Справка о внедрении практических рекомендаций ВКР

Справка о внедрении практических рекомендаций ВКР не является обязательным документом для ее защиты на заседании аттестационной комиссии. Однако ее наличие характеризует высокий уровень выполнения ВКР и готовность выпускника квалифицированно решать профессиональные задачи.

В образовательной организации поощряется представление на защиту справок о внедрении рекомендаций ВКР в практику работы конкретного предприятия (организации, фирмы и т.п.). В первую очередь это относится к предприятию, на базе которого выполнялась ВКР.

Справка пишется в произвольной форме, но с обязательным указанием конкретных практических рекомендаций, которые автор ВКР внедрил в работу предприятия с указанием конкретного места (участка, цеха, подразделения, службы, отдела и т.п.), где эти рекомендации были применены.

Справка прилагается к ВКР и представляется в аттестационную комиссию.

Образец справки о внедрении приводится в Приложении М.

### 4.3 Процедура и результаты публичной защиты ВКР

Процедура защиты ВКР определяется Положением об итоговой государственной аттестации выпускников высших учебных заведений Российской Федерации, утвержденным приказом Минобразования России от 25.03.2003 № 1155, и проводится в соответствии с Порядком проведения защиты ВКР.

В соответствии с Положением к защите ВКР допускается обучающийся, успешно завершивший в полном объеме освоение основной образовательной программы высшего образования по направлению подготовки (специальности) высшего образования, разработанной образовательной организации в соответствии с требованиями ФГОС и ГОС ВПО, и успешно прошедший установленные итоговые экзамены.

Защита ВКР проходит в торжественной обстановке, публично, на открытом заседании аттестационной комиссии. Выпускнику заранее предоставляется информация о дате, времени и месте работы комиссии, которые отражены в расписании, утвержденном ректором.

В начале работы комиссии Председатель представляет выпускникам и присутствующим всех ее членов, с указанием фамилии, имени и отчества, ученой степени и звания, должности, которую они занимают.

Объявляя защиту каждой ВКР, Председатель называет фамилию, имя и обязательно отчество докладчика, тему ВКР, а также время, отводимое на доклад. Члены комиссии, задавая вопросы, также обращаются к выпускникам по имени и отчеству.

Продолжительность защиты выпускной квалификационной работы – 15-20 минут.

Схематично процедура защиты включает следующие стадии.

1. Доклад выпускника по теме ВКР.

2. Ответы на вопросы членов комиссии.

3. Оглашение рецензии на выпускную квалификационную работу (для специалитета).

4. Ответы выпускника на замечания рецензента (для специалитета)..

5. Выступление руководителя ВКР и других лиц, присутствующих на защите, если они просят слово.

6. Ответы выпускника на критические замечания руководителя и других лиц, принимающих участие в обсуждении ВКР.

После заслушивания докладов всех или части выпускников, представляемых ВКР на защиту, проводится закрытое заседание аттестационной комиссии. На нем обсуждаются результаты защиты каждого выпускника по определенным критериям, выносится итоговая оценка каждому выпускнику: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Критериями оценки качества ВКР являются:

а) творческий и самостоятельный подход выпускника к разработке темы ВКР;

б) научный уровень проведенного исследования темы ВКР;

в) глубина и оригинальность анализа источников и научной литературы;

г) умение систематизировать и обобщать информацию, самостоятельно решать поставленные в ВКР цели и задачи (в том числе нестандартные) с использованием передовых научных методик и технологий;

д) систематичность, логичность и завершенность изложения основных положений и результатов ВКР;

е) научная обоснованность теоретических выводов и практических рекомендаций;

ж) оформление ВКР в соответствии с действующими ГОСТами и методическими указаниями образовательной организации;

и) степень обладания общими и профессиональными компетенциями, проявившимися как в содержании ВКР, так и в процессе ее защиты.

Итоговая оценка определяется простым большинством голосов членов комиссии, участвующих в заседании (при равенстве голосов решающим является голос Председателя комиссии). Одновременно принимаются рекомендации о практическом использовании полученных в ВКР результатов.

5. После окончания закрытого заседания аттестационной комиссии возобновляется открытое заседание, на которое вместе с выпускниками приглашаются все желающие. Председатель кратко подводит итоги защиты, объявляет оценки по защищенным на данном заседании ВКР.

6. Решения аттестационной комиссии об оценке качества ВКР выпускников оформляются протоколами установленной формы.

**ГЛОССАРИЙ**

| № п/п | Новое понятие | Содержание |
|---|---|---|
| 1 | **IP-хелпинг** | индивидуальное асинхронное консультирование, осуществляемое в сети Интернет в открытой автоматизированной информационной системе (ОАЗИС), в процессе которого обучающийся задаёт вопросы руководителю ВКР (преподавателю учебной дисциплины), а руководитель ВКР (преподаватель) готовит ответы на специальном сайте образовательной организации |
| 2 | **Базовый вуз** | образовательное учреждение высшего образования, реализующее полный цикл обучения и осуществляющее организационное, научное и методическое обеспечение учебного процесса в своих территориальных структурных подразделениях |
| 3 | **Бакалавр** | квалификация (степень), присваиваемая выпускнику высшего учебного заведения, освоившему первую ступень высшего образования, успешно прошедшему итоговую аттестацию и защитившему выпускную квалификационную работу |
| 4 | **Виртуальный читальный зал** | рабочие места обучающихся, оборудованные персональными компьютерами с постоянными IP-адресами, имеющие регистрацию в ТКДБ и позволяющие в сети Интернет изучать информационные ресурсы электронных библиотек, внешних информационных баз образовательной и культурной направленности, а также коллекции электронных журналов |
| 5 | **Выпускная квалификационная работа** | завершенная научно-квалификационная учебно-исследовательская работа выпускника вуза по определенной теме (проблеме), направленная на систематизацию, закрепление и расширение у него знаний, формирование и развитие навыков и умений самостоятельного решения конкретных научных задач, характеризующая итоговый уровень квалификации и подтверждающая готовность к профессиональной деятельности |
| 6 | **Выпускник** | лицо, успешно завершившее теоретическое и практическое обучение по основной образовательной программе и приказом допущенное к итоговой аттестации |
| 7 | **Глоссарий** | толковый (объясняющий) словарь понятий и терминов |
| 8 | **Диплом** | документ, который выдается выпускникам образовательных учреждений, имеющих государственную аккредитацию, лицам, прошедшим государственную (итоговую) аттестацию, свидетельствующий об окончании образовательного учреждения ВЫСШЕГО ОБРАЗОВАНИЯ или среднего профессионального образования и присвоении соответствующей квалификации и ученой степени |
| 9 | **Дипломированный специалист** | квалификация (степень), присваиваемая выпускнику высшего учебного заведения, освоившему ступень ВЫСШЕГО ОБРАЗОВАНИЯ , успешно прошедшему итоговую аттестацию и защитившему дипломную работу (проект) |
| 10 | **Информационные ресурсы** | совокупность данных, организованных для эффективного получения достоверной информации |
| 11 | **ИР КОП** | интеллектуальный робот контроля оригинальности и профессионализма |
| 12 | **Итоговая аттестация** | комплексная оценка уровня подготовки выпускника образовательного учреждения на соответствие требованиям федерального государственного образовательного стандарта |
| 13 | **Личная студия** | сайт, на котором обучающийся может работать с учебными продуктами по дисциплинам, входящими в его индивидуальный учебный план |
| 14 | **Монография** | научное исследование, посвященное одному вопросу, проблеме, теме |
| | **Нормоконтроль** | процедура, которая проводится в образовательной организации с целью обеспечения единообразия структуры и оформления курсовых работ и выпускных квалификационных работ |
| 15 | **Отзыв** | оценивание руководителем ВКР проведенной научно- |

| № п/п | Новое понятие | Содержание |
|---|---|---|
| | | исследовательской работы выпускника, с отражением актуальности темы, направленности исследования, и указанием ценности проведенного исследования |
| 16 | **Публикация** | документ, доступный для массового использования |
| 17 | **Рецензия** | отзыв на ВКР, написанный другим специалистом (кроме руководителя ВКР) |
| 18 | **Самостоятельная работа обучающегося** | разновидность учебной деятельности обучающихся, направленная на выполнение различных заданий учебного, исследовательского и самообразовательного характера;<br>способ усвоения системы знаний, навыков и умений, познавательной деятельности обучающихся |
| 19 | **Слайд-тьюторинг** | учебное занятие по подготовке обучающихся к экзаменам, выполнению курсовой работы, выпускной квалификационной работы, заданий практик в форме индивидуального или коллективного просмотра обучающимися видеозаписей телевизионных консультаций преподавателей (руководителей выпускной квалификационной работы) |
| 20 | **Телекоммуникационная двухуровневая библиотека (ТКДБ)** | организованное хранилище публикаций, предназначенное для быстрого поиска и доступа обучающихся к изданиям учебной, учебно-методической, научной и справочной литературы на электронном (цифровом) носителе |
| 21 | **Федеральный государственный образовательный стандарт** | нормативный правовой акт, устанавливающий совокупность обязательных требований к образованию определенного уровня и (или) профессии, специальности и направления подготовки |
| 22 | **Электронное обучение (e-learning)** | образовательный процесс с применением содержащейся в базах данных и используемой при реализации образовательных программ информации и обеспечивающих ее обработку информационных технологий, технических средств, а также информационно-телекоммуникационных сетей, обеспечивающих передачу по линиям связи указанной информации, взаимодействие участников образовательного процесса |

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

### Нормативные правовые акты


1. Об образовании [Текст] : Федеральный закон РФ от 29 декабря 2012 г. № 273-ФЗ (с изм. и доп., вступ. в силу с 03.07.2016 г.) // Собр. законодательства Рос. Федерации. – 2012. – № 53. – Ст. 7598.
2. Порядок организации и осуществления образовательной деятельности по образовательным программам высшего образования - программам бакалавриата, программам специалитета, программам магистратуры, утвержденный Приказом Минобрнауки России от 19.12.2013 № 1367. (Зарегистрировано в Минюсте России 24.02.2014 N 31402).
3. Порядок применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ, утвержденный Приказом Минобрнауки России от 09.01.2014 № 2. (Зарегистрировано в Минюсте России 04.04.2014 N 31823)
4. Положение об итоговой государственной аттестации выпускников высших учебных заведений в Российской Федерации [Текст] : Приказ Минобразования РФ от 25 марта 2003 г. № 1155 // Рос. газета. – 2003.
5. Положение об итоговой аттестации [Текст] : инструкция (утв. решением Ученого совета СГА, 2011).
6. Порядок проведения защиты выпускной квалификационной работы [Текст] : инструкция (утв. решением Ученого совета СГА, 2011).

7. Порядок проведения итоговой аттестации [Текст] : инструкция (утв. решением Ученого совета СГА, 2012).

**Учебные издания**

1. Карпенко, М. П. Телеобучение [Текст] / М. П. Карпенко. – М. : СГА, 2008. – 800 с.

**ПРИЛОЖЕНИЕ А Задание на выполнение ВКР**

Форма 11-д

**З А Д А Н И Е**
**на выполнение выпускной квалификационной работы**

Выпускная квалификационная работа (ВКР) выполнена в форме:

Бакалаврской работы ☐ Магистерской диссертации ☐

Дипломной работы ☐ Дипломного проекта ☐

Студент(ка)_____
фамилия, имя, отчество

форма обучения _____, № контракта _____, группа_____,
очная/заочная/очно-заочная (вечерняя)

направление подготовки / специальность _____
нужное подчеркнуть наименование

1 Тема _____

_____

_____

2 Дата выдачи темы «_____»_____201__г.

3 Календарный график выполнения _____

_____

_____

4 Содержание пояснительной записки _____

_____

_____

5 Срок представления студентом(кой) законченной ВКР:

«___» _____ 201 __г.

Руководитель _____
Ф.И.О., ученая степень, должность, место работы

_____
_____

Научный руководитель _____
(подпись)

Студент(ка) _____
(подпись)

**ПРИЛОЖЕНИЕ Б Унифицированные требования**

Унифицированные требования к оформлению выпускных квалификационных работ

| № п/п | Объект унификации | Параметры унификации | |
|---|---|---|---|
| | | бакалаврская работа | дипломная работа (проект) |
| 1 | Формат листа бумаги | А4 | А4 |
| 2 | Размер шрифта | 14 пунктов | 14 пунктов |
| 3 | Название шрифта | Times New Roman | Times New Roman |
| 4 | Междустрочный интервал | Полуторный | Полуторный |
| 5 | Количество строк на странице | 28-30 строк<br>(1800 печатных знаков) | 28-30 строк<br>(1800 печатных знаков) |
| 6 | Абзац | 1,25 см (5 знаков) | 1,25 см (5 знаков) |

| № п/п | Объект унификации | Параметры унификации | |
|---|---|---|---|
| | | бакалаврская работа | дипломная работа (проект) |
| 7 | Поля (мм) | Левое, верхнее и нижнее – 20, правое – 10 | Левое, верхнее и нижнее – 20, правое – 10. |
| 8 | Общий объем без приложений | 50-70 стр. машинописного текста | 90-110 стр. машинописного текста |
| 9 | Объем введения | 2-4 стр. машинописного текста | 5-7 стр. машинописного текста |
| 10 | Объем основной части | 40-50 стр. машинописного текста | 80-100 стр. машинописного текста |
| 11 | Объем заключения | 3-5 стр. машинописного текста (примерно равен объему введения) | 5-7 стр. машинописного текста (примерно равен объему введения) |
| 12 | Нумерация страниц | Сквозная, в нижней части листа, посередине. На титульном листе номер страницы не проставляется | Сквозная, в нижней части листа, посередине. На титульном листе номер страницы не проставляется |
| 13 | Последовательность приведения структурных частей работы | Титульный лист. Задание на выполнение выпускной квалификационной работы. Содержание. Введение. Основная часть. Заключение. Глоссарий. Список использованных источников. Список сокращений. Приложения | Титульный лист. Задание на выполнение выпускной квалификационной работы. Содержание. Введение. Основная часть. Заключение. Глоссарий. Список использованных источников. Список сокращений. Приложения |
| 14 | Оформление структурных частей работы | Каждая структурная часть начинается с новой страницы. Наименования приводятся с абзаца с прописной (заглавной буквы). Точка в конце наименования не ставится | Каждая структурная часть начинается с новой страницы. Наименования приводятся с абзаца с прописной (заглавной буквы). Точка в конце наименования не ставится |
| 15 | Структура основной части | 2-3 главы, соразмерные по объему | 3 главы, соразмерные по объему |
| 16 | Наличие глоссария | Обязательно.<br>15-20 понятий | Обязательно.<br>Не менее 25 понятий |
| 17 | Состав списка использованных источников | 30-50 библиографических описаний документальных и литературных источников | Не менее 50 библиографических описаний документальных и литературных источников |
| 18 | Наличие приложений | Обязательно | Обязательно |
| 19 | Оформление содержания | Содержание включает в себя заголовки всех разделов, глав, параграфов, глоссария, приложений с указанием страниц начала каждой части | Содержание включает в себя заголовки всех разделов, глав, параграфов, глоссария, приложений с указанием страниц начала каждой части |

**ПРИЛОЖЕНИЕ В Пример содержания* ВКР бакалавра юриспруденции**

**Содержание**



* Направление подготовки 030500.62 – «Юриспруденция», степень (квалификация) – бакалавр юриспруденции.

Тема: «Правовое регулирование несостоятельности (банкротства) в Российской Федерации».

**ПРИЛОЖЕНИЕ Г Пример содержания* ВКР специалиста менеджмента**

*Содержание*



* Направление подготовки 080500.65 – «Менеджмент», степень (квалификация) – специалист менеджмента.

Тема: «Роль и методы стратегического менеджмента и его использование в практике формирования и достижения важнейших целей развития предприятия».

**ПРИЛОЖЕНИЕ Д Отзыв**

Форма 19-д/7

**ОТЗЫВ**
**на выпускную квалификационную работу**

Студента(ки) _____
фамилия, имя, отчество

на тему _____

_____

_____

Выпускная квалификационная работа (ВКР) выполнена в форме:

Бакалаврской работы ☐ Дипломной работы ☐ Дипломного проекта ☐

1 Актуальность и практическая значимость темы _____

2 Логическая последовательность

3 Аргументированность и конкретность выводов и предложений _____

_____

_____4 Правильное

использование научных/профессиональных терминов и понятий в контексте проблемы

_____

_____

5 Уровень использования различных видов литературных источников _____

6 Качество оформления ВКР, качество таблиц, иллюстраций и пр.

_____

7 Уровень самостоятельности при работе над темой ВКР _____

_____

_____

_____

_____

_____

8 Недостатки работы _____

_____

_____

_____

_____

_____

9 Предложения для внедрения _____

_____

_____

_____

10 ВКР соответствует/не соответствует требованиям, предъявляемым к ВКР, и может/не может

(нужное подчеркнуть)

(нужное подчеркнуть)

быть рекомендована к защите на заседании Государственной аттестационной комиссии

11 Студент(ка) _____

фамилия, имя, отчество

заслуживает присвоения ему (ей) степени _____

бакалавра, специалиста, (вписать нужное)

по направлению подготовки / специальности _____

Руководитель ВКР _____

фамилия, и., о., ученая степень, звание, место работы, должность

_____

_____

«_____»_____201__г. _____

подпись руководителя

**ПРИЛОЖЕНИЕ Е Титульный лист**

Форма 09-д

Направление подготовки / специальность

_____

ДОПУСК К ЗАЩИТЕ:

Приказ № _____

от «____»_____201__г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Вид ВКР** бакалаврская работа

Тема: _____

_____

_____

Обучающийся _____ / _____ /
Ф. И. О. подпись

№ контракта _____ Группа _____

Руководитель: _____ / _____ /
Ф. И. О. подпись

Дата представления работы «____» _____ 201__г.

Москва 201 __ г.

**ПРИЛОЖЕНИЕ Ж Титульный лист для демонстрационных материалов**

Форма ДМ-01

**Демонстрационный материал***
**к выпускной квалификационной работе**

Выпускная квалификационная работа выполнена в форме:

Бакалаврской работы ☐ Магистерской диссертации ☐

Дипломной работы ☐ Дипломного проекта ☐

Демонстрационный материал оформлен в виде:

«Раздаточного материала» ☐ Плакатов ☐

Студент(ка) _____
фамилия, имя, отчество

форма обучения _____, № контракта _____, группа _____,
очная/заочная

направление подготовки / специальность _____
нужное подчеркнуть наименование

1 Тема _____

_____2 Руководитель

ВКР _____
фамилия, и.о., ученая степень, звание

_____

3 «Раздаточный материал» / плакаты _____
количество листов

4 Перечень листов 1 _____

2 _____

3 _____

4 _____

5 _____

6 _____

7 _____

Студент(ка) _____
(подпись)

Руководитель ВКР _____/_____/
(подпись) (расшифровка подписи)

* «Раздаточный материал» к ВКР оформляется выпускником и утверждается руководителем ВКР. Представляется выпускником членам ГАК перед защитой ВКР.

**ПРИЛОЖЕНИЕ И Информация для демонстрации на защите**

**Примерный состав информации, представляемой на демонстрационных плакатах (в «раздаточном материале») на защите выпускной квалификационной работы**

1. Цель и задачи выполнения выпускной квалификационной работы, в том числе изображенные в виде дерева целей.
2. Таблицы, диаграммы и графики, блок-схемы, характеризующие объект исследования.
3. Методика исследования.
4. Результаты, полученные при выполнении выпускной квалификационной работы.
5. Рекомендации по внедрению в практику деятельности предприятия (организации, фирмы) результатов выпускной квалификационной работы.
6. Данные из справки о внедрении результатов выпускной квалификационной работы на предприятии (организации, фирме).

***Примечание: общее количество демонстрационных плакатов 4-6 листов; общее количество информационных страниц, приводимых в «раздаточном материале», 8-10 страниц.***

**ПРИЛОЖЕНИЕ К Рекомендации к докладу по защите ВКР**

**Схема доклада по защите выпускной квалификационной работы**

1. ***Обращение: Уважаемые члены Государственной аттестационной комиссии! Вашему вниманию предлагается выпускная квалификационная работа на тему...***

2. В 2-3 предложениях дается характеристика актуальности темы.

3. Приводится краткий обзор литературных источников по избранной проблеме (степень разработанности проблемы).

4. ***Цель выпускной квалификационной работы*** - формулируется цель бакалаврской работы.

5. Формулируются задачи, приводятся названия глав. При этом в формулировке должны присутствовать глаголы типа - изучить, рассмотреть, раскрыть, сформулировать, проанали-зировать, определить и т.п.

6. Из каждой главы используются выводы или формулировки, характеризующие результаты. Здесь можно демонстрировать плакаты (раздаточный материал). При демонстрации плакатов не следует читать текст, изображенный на них. Надо только описать изображение в одной-двух фразах. Если демонстрируются графики, то их надо назвать и констатировать тенденции, просматриваемые на графиках. При демонстрации диаграмм обратить внимание на обозначение сегментов, столбцов и т.п. Графический материал должен быть наглядным и понятным со стороны. Текст, сопровождающий диаграммы и гистограммы, должен отражать лишь конкретные выводы. Объем этой части доклада не должен превышать 1,5-2 стр. печатного текста.

7. ***В результате проведенного исследования были сделаны следующие выводы:*** (формулируются основные выводы, вынесенные в заключение).

8. ***Опираясь на выводы, были сделаны следующие предложения:*** (перечисляются предложения).

*Примечание. Седьмая и восьмая части доклада не должны превышать в сумме 1 стр. печатного текста.*

Всего весь доклад с хронометражем в 10-12 минут (с демонстрационным материалом) укладывается на 3-4 стр. печатного текста с междустрочным интервалом 1,0 и шрифтом (14 пунктов).

Завершается доклад словами: ***Благодарю за внимание***.

**ПРИЛОЖЕНИЕ Л Образец рецензии на ВКР**

Рецензия на выпускную квалификационную работу
студента Иванова Сергея Александровича

Содержание выпускной квалификационной работы Иванова Сергея Александровича: «Совершенствование оценки инновационной деятельности на предприятии (на примере ОАО «Прогресс»)» соответствует утвержденной теме и является актуальной для предприятия, по материалам которого выполнялась.

В выпускной квалификационной работе наиболее полно освещены разделы, связанные с разработкой методических вопросов по планированию инноваций в ОАО «Прогресс». В них автор предложил усовершенствовать действующий в ОАО «Прогресс» порядок планирования инноваций, на основе дополнительного учета экологических факторов. Это позволяет говорить о наличии в выпускной квалификационной работе самостоятельных и оригинальных решений.

К достоинствам выпускной квалификационной работы можно также отнести: *(перечисляются достоинства работы)*

Учитывая вышеизложенное, можно утверждать о практической значимости для ОАО «Прогресс» результатов, полученных в рецензируемой выпускной квалификационной работе.

Вместе с тем в работе Иванова С.А. выявлены недостатки: *(перечисляются недостатки работ)* Представленные на просмотр 4 демонстрационных плаката полностью соответствуют графическим материалам в тексте выпускной квалификационной работы и согласуются с результатами анализируемых Ивановым С.А. экономических и управленческих процессов.

Качество оформления выпускной квалификационной работы является достаточно высоким. Пояснительная записка набрана на компьютере, грамотно написана, тщательно вычитана, грамматические и синтаксические ошибки и опечатки отсутствуют.

Полученные студентом Ивановым С.А. в образовательной организации теоретические знания и умения можно считать достаточными для его самостоятельной трудовой деятельности на должностях, требующих высшего экономического образования.

Выпускная квалификационная работа студента Иванова С.А. заслуживает оценки «отлично», а он сам – присвоения искомой квалификации «экономист».

канд. экон. наук, доц. (должность) С.А. Феоктистов

Подпись С.А. Феоктистова заверяю:

Начальник отдела кадров М. А. Кедров

ПЕЧАТЬ

**ПРИЛОЖЕНИЕ М Образец справки о внедрении результатов ВКР**

СПРАВКА
о внедрении рекомендаций, разработанных в выпускной квалификационной работе
обучающегося в Иванова Сергея Александровича

***В процессе выполнения выпускной квалификационной работы на тему: «Совершенствование оценки инновационной деятельности на предприятии (на примере ОАО «Прогресс»)» студент Иванов С.А. принимал участие в разработке*** **(перечисляются разработанные вопросы)**.

***Полученные им результаты, включающие в себя*** **(перечисляется то, что конкретно сделано студентом)**.

***нашли отражение в методических разработках по планированию инноваций в ОАО «Прогресс»*** **(либо в докладных, аналитических и прочих записках, направленных в Совет директоров ОАО «Прогресс» (другой руководящий орган), либо использованы в расчетах эффективности инноваций в ОАО «Прогресс» и т.п.).**

***В настоящее время указанные методические разработки распоряжением директора по экономике и финансам ОАО «Прогресс» (№ _____ от 5 марта 2001 г.) включены в инструктивные материалы, которыми должны руководствоваться работники отдела новых технологий ОАО.***

***Генеральный директор С.П. Кошелев***

ПЕЧАТЬ

*(На крупных предприятиях (организациях, фирмах) справка может быть также подписана начальником департамента, отдела, цеха или другого структурного подразделения.
В таких случаях подпись специалиста заверяется руководителем отдела кадров (канцелярии) и соответствующей печатью)*

**УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

ПОРЯДОК НАПИСАНИЯ, ОФОРМЛЕНИЯ И ЗАЩИТЫ

Ответственный за выпуск Е.Д. Кожевникова
Корректор Н.Н. Горбатова
Оператор компьютерной верстки С.А. Кафтанников