Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра общей математики и информатики

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ДИСЦИПЛИНЫ «ПРИКЛАДНАЯ ИНФОРМАТИКА»

основной образовательной программы по специальности 160400.65 – проектирование, производство и эксплуатация ракет и ракетно-космических комплексов

Благовещенск 2013

УМКД разработан канд. пед. наук, доцентом, Чалкиной Натальей Анатольевной

Рассмотрен и рекомендован на заседании кафедры

Протокол заседания кафедры от «__15__»_мая__2013 г., № _9__

Зав. кафедрой _____/ Г.В. Литовка /
(подпись)

УТВЕРЖДЕН

Протокол заседания УМСС 160400.65 – проектирование, производство и эксплуатация ракет и ракетно-космических комплексов

от «___»_____201__ г., №___

Председатель _____/ _____/
(подпись) (И.О. Фамилия)

СОДЕРЖАНИЕ

I. РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Цель дисциплины: воспитание у студентов информационной культуры; высокого уровня квалификации в направлениях технического обеспечения, системных и прикладных программных средств, программирования.

Задачи дисциплины:

– углубление знаний студентов по основному аппаратному обеспечению и периферийным устройствам компьютера;

– научить студентов решать задачи, возникающие в процессе сопровождения и эксплуатации программных средств;

– овладение студентами современными методами и средствами программирования.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП ВПО

Предлагаемая дисциплина относится к базовой части математического и естественнонаучного цикла ООП.

Для успешного освоения данной дисциплины необходимы базовые знания курса «Информатика» в объеме средней общеобразовательной школы.

Дисциплина занимает важное место в программе подготовки бакалавра, так как обеспечивает базовую подготовку студентов в области использования средств вычислительной техники: для всех курсов, использующих автоматизированные методы анализа, расчетов и компьютерного оформления курсовых и дипломных проектов.

3. КОМПЕТЕНЦИИ ОБУЧАЮЩЕГОСЯ, ФОРМИРУЕМЫЕ В РЕЗУЛЬТАТЕ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

В процессе освоения данной дисциплины студент формирует и демонстрирует следующие общеобразовательные компетенции:

– способен использовать базовые положения математики, естественных, гуманитарных и экономических наук при решении социальных и профессиональных задач, способен критически оценивать основные теории и концепции, границы их применения (ОК-2);

– имеет навыки работы с компьютером как средством управления, в том числе в режиме удаленного доступа, готов работать с программными средствами общего и специального назначения (ОК-14);

– способен использовать в профессиональной деятельности знания и методы, полученные при изучении математических и естественнонаучных дисциплин (ПК-1);

– понимает роль математических и естественнонаучных наук и способность к приобретению новых математических и естественно-научных знаний, с использованием современных образовательных и информационных технологий (ПК-4);

– самостоятельно разрабатывает, с помощью алгоритмических языков, программы для исследования процессов, описанных математическими моделями (ПК-14).

В результате освоения обучающийся должен демонстрировать следующие результаты образования:

1) Знать: основные положения информатики, понятие о двоичной системе. Методы составления алгоритмов, основные математические языки программирования. Принцы работы с клавиатурой;

2) Уметь: программировать на одном из алгоритмических языков; работать в современной программной среде и Интернете.

3) Владеть: методами составления программ для решения задач на ЭВМ.

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины составляет 3 зачетных единицы, 108 часов.

№ п/п	Раздел дисциплины	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)				Формы текущего контроля успеваемости Форма промежуточной аттестации
				Лекции	Лабораторные работы	Практические занятия	Самостоятельная работа	
8	Технология программирования задач обработки строковых и графических данных	3	1-9	6	6	14	20	тест
9	Базы данных	3	10-18	10	10	4	38	тест
	ИТОГО	3		16	16	18	58	зачет

5. СОДЕРЖАНИЕ РАЗДЕЛОВ И ТЕМ ДИСЦИПЛИНЫ

5.1. Лекции

№ п/п	Наименование темы	Содержание темы
1	Технология программирования задач обработки строковых и графических данных	Строковый тип данных. Процедуры и функции для работы со строковыми данными. Программирование задач по обработке графических данных
2	Базы данных	Понятие базы данных. Модели организации данных. Язык SQL. Системы управления базами данных. Основные понятия СУБД Access: поле данных, ключ поля данных, схема данных, таблицы, формы, запросы, отчеты.

5.2. Лабораторные занятия

Наименование темы	Содержание темы
Программирование задач со строковыми данными	Поиск заданных символов в тексте. Замена символов в тексте.
Программы построения графиков	Компонент построения графиков TChart. Построение различных графиков.
Страница Диалоги	Основные компоненты страницы Dialogs. Создание меню, подпунктов меню
Базы данных	Создание таблиц различными способами. Схема данных. Работа с запросами. Создание форм и отчетов.

5.3 Практические занятия

Наименование темы	Содержание темы
Обработка строковых данных	Построение блок-схем для задач обработки строковых данных
Обработка графиков	Построение блок-схем для задач по обработке графических данных
Базы данных	Создание таблиц различными способами. Схема данных. Работа с запросами. Создание форм и отчетов.

6. САМОСТОЯТЕЛЬНАЯ РАБОТА

№ п/п	№ раздела (темы) дисциплины	Форма (вид) самостоятельной работы	Трудоемкость в часах
1	1	Подготовка к лабораторной работе с использованием обучающего теста. Выполнение лабораторных работ	20
2	2	Подготовка к лабораторной работе с использованием обучающего теста. Выполнение лабораторных работ	38

7. МАТРИЦА КОМПЕТЕНЦИЙ УЧЕБНОЙ ДИСЦИПЛИНЫ

Тема дисциплины	Компетенции					ИТОГО
	ОК-2	ОК-14	ПК-1	ПК-4	ПК-14	
Тема 1	+	+	+	+	+	5
Тема 2	+	+	+	+	+	5

8. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ И ФОРМЫ

Образовательный процесс по дисциплине строится на основе комбинации следующих образовательных технологий.

Интегральную модель образовательного процесса по дисциплине формируют технологии методологического уровня: модульно-рейтинговое обучение, технология поэтапного формирования умственных действий, технология развивающего обучения, элементы технологии развития критического мышления.

Удельный вес занятий, проводимых в интерактивных формах, составляет 30% аудиторных занятий (15 ч).

Тема	Вид занятия	Кол-во часов
Технология программирования задач обработки строковых и графических данных	Мозговой штурм	7
Базы данных	Метод проектов	8
		15

Рекомендуется использование информационных технологий при организации коммуникации со студентами для представления информации, выдачи рекомендаций и консультирования по оперативным вопросам (электронная почта), использование мультимедиа-средств при проведении лекционных и лабораторных занятий.

9. ОЦЕНОЧНЫЕ СРЕДСТВА ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

В качестве основных средств текущего контроля используется тестирование. В качестве дополнительной формы текущего контроля предлагаются аудиторные и внеаудиторные письменные задания (контрольные работы).

Для самостоятельной работы используется учебно-методическое обеспечение на бумажных и электронных носителях. Тематика самостоятельной работы соответствует содержанию

разделов дисциплины и теме домашнего задания. Освоение материала контролируется в процессе проведения лабораторных занятий.

Контрольные вопросы и задания для проведения текущего контроля выбираются из содержания разделов дисциплины. Выполнение домашнего задания обеспечивает непрерывный контроль за процессом освоения учебного материала каждого обучающегося, своевременное выявление и устранение отставаний и ошибок.

Промежуточная аттестация по итогам освоения дисциплины: зачет.

Вопросы к зачету (3 семестр)

1. Базы данных. Модели базы данных.
2. Базы данных. Типы данных.
3. Структурные элементы базы данных.
4. Базы данных. Таблицы.
5. Базы данных. Запросы.
6. Базы данных Отчеты.
7. Базы данных. Формы.
8. Базы данных. Макросы.
9. Базы данных. Модули.
10. Базы данных. Страницы.
11. Типы связей в базе данных.
12. Система управления базами данных.
13. Циклические структуры в Object Free Pascal.
14. Язык программирования Object Free Pascal. Данные. Типы данных языка.
15. Язык программирования Object Free Pascal. Арифметические операции и функции.
16. Компонент построения графиков TChart.
17. Программирование задач на построение графиков.

10. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

а) основная литература:

1. . Информатика: учеб.: рек. Мин. обр. РФ / под ред. Н. В. Макаровой. – М.: Финансы и статистика, 2005, 2007. – 268 с.

2. Информатика: учеб.: рек. Мин. обр. РФ / под ред. Н. В. Макаровой. – 3-е изд., перераб. – М.: Финансы и статистика, 2009. – 768 с.

3. Каймин В.А. Информатика: учеб.: рек. Мин. обр. РФ / В.А. Каймин. – 5-е изд. – М.: Инфра-М, 2008. – 285 с.

4. Острейковский, В. А. Информатика: Учеб.: рек. Мин. обр. РФ / В.А. Острейковский. – 5-е изд., стер. – М.: Высш. шк., 2009. – 512 с.

б) дополнительная литература:

1. Безручко В.Т. Информатика (курс лекций): учеб. пособие: рек. НМС / В.Т. Безручко. – М.: ФОРУМ: ИНФРА-М, 2006. – 432 с.

2. Информатика и программирование: компьютерный практикум: учеб. Пособие: рек УМО / А.Н. Гуда [и др.]; под общ. ред. В.И. Колесникова. – М.: Дашков и К, 2009. – 238 с.

3. Могилев А. В. Информатика: учеб. пособие: рек. Мин. обр. РФ / А.В. Могилев, Е.К. Хеннер, Н.И. Пак; под ред. А.В. Могилева. – 2-е изд., стер. – М.: Академия, 2008. – 328 с.

4. Степанов А.Н. Информатика: базовый курс для студентов гуманитар. спец. высш. учеб. заведений / А.Н. Степанов. – 6-е. изд. – СПб.: Питер, 2010. – 720 с.

5. Шапорев С.Д. Информатика. Теоретический курс и практические занятия: учеб.: рек. НМС / С.Д. Шапорев. – СПб.: БХВ-Петербург, 2008. – 469 с.

в) перечень журналов:

1. Информационные технологии.
2. Информационные технологии и вычислительные системы.
3. Информатика и системы управления.

г) программное обеспечение и Интернет-ресурсы

№	Наименование ресурса	Краткая характеристика
1	http://informatka.ru/	Содержит справочный материал по различным разделам информатики.
2	http://www.iqlib.ru	Интернет-библиотека образовательных изданий, в которой собраны электронные учебники, справочные и учебные пособия. Удобный поиск по ключевым словам отдельным темам и отраслям знаний
3	http://elibrary.ru	Научная электронная библиотека журналов
4	Windows	Операционная система
5	Total Commander, Far Manager	Операционная оболочка
6	Microsoft Office	Пакет прикладных программ
7	WinZip,WinRAR 3.2	Программа-архиватор
8	Delphi for Object Pascal, Lazarus	Система объектно-ориентированного программирования

11. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Класс ПЭВМ на базе процессора Intel Pentium.

12. РЕЙТИНГ-ПЛАН ДИСЦИПЛИНЫ

Модуль	**Название**	**Кол. баллов за модуль**	**Темы**	**Кол. баллов за тему**	**Виды работ**
1	Технология программирования задач обработки строковых и графических данных	45	Обработка строковых данных	15	Лаб.раб., тест
			Построение графиков	15	Лаб.раб.
			Страница Dialogs	15	Лаб.раб.
2	Базы данных	15		15	Лаб. раб., тест
	Зачет	40			
	Итог	***100***			

II. КРАТКОЕ ИЗЛОЖЕНИЕ ПРОГРАММНОГО МАТЕРИАЛА

Тема 1. Технология программирования задач обработки строковых и графических данных

План лекции:

1. Строковый тип данных.
2. Процедуры и функции для работы со строковыми данными.
3. Программирование задач по обработке графических данных

Цель: формирование знаний об основных вычислительных алгоритмах.

Задачи:

– развитие практических навыков решения типовых задач алгоритмизации;

– формирование навыка записи алгоритма на неформальном языке и перевода алгоритма на язык практического программирования.

Ключевые вопросы:

1. Укажите название строкового типа данных.
2. Определите назначение использования квадратных скобок при описании типа String[n]. О чем говорит отсутствие данных скобок?
3. Какие знаки используются для задания строки?
4. Как выглядит обращение к символу строки?
5. Укажите знак, используемый для сложения строк.
6. Укажите функцию отыскания позиции подстроки в строке. Какого типа данных результат данной функции?
7. Запишите функцию длины строки.
8. Назовите форматы функций удаления, копирования, вставки подстроки в строке.

Рекомендуемая литература:

1. Информатика и программирование: компьютерный практикум: учеб. пособие: рек УМО / А.Н. Гуда [и др.]; под общ. ред. В.И. Колесникова. – М.: Дашков и К, 2009. – 238 с.

Тема 2. Базы данных

План лекции:

1. Понятие базы данных.
2. Модели организации данных. Язык SQL.
3. Системы управления базами данных.
4. Основные понятия СУБД Access: поле данных, ключ поля данных, схема данных, таблицы, формы, запросы, отчеты.

Цель: познакомиться с интерфейсом базы данных; научиться создавать ключевые поля, устанавливать связи между таблицами; приобрести опыт удаления и восстановления информации из связанных таблиц.

Задачи:

– обучить основам создания и ведения баз данных;

– научить создавать отчеты, формы, запросы к базам данных.

Ключевые вопросы:

1. Что такое база данных?
2. Классификация баз данных.
3. В каких объектах хранятся данные базы?
4. Какую базу данных называют реляционной?
5. Для чего предназначены запросы?
6. Какое поле можно считать уникальным?
7. Чем отличаются поля и записи таблицы?
8. В чем состоит особенность поля Счетчик?
9. В каком диалоговом окне создаются связи между полями таблиц базы данных?
10. Для чего предназначены запросы?
11. Какие итоговые функции вы знаете?
12. Для чего предназначены формы?

13. Какие методы автоматического создания форм вы знаете?

14. Для чего предназначены отчеты?

15. Что общего и в чем различие между разделами отчетов и разделами форм?

16. Можно ли использовать формы не только для ввода, но и для вывода данных? Если да, то на какое устройство компьютерной системы выполняется этот вывод?

Рекомендуемая литература:

1. Информатика: учеб.: рек. Мин. обр. РФ / под ред. Н. В. Макаровой. – М.: Финансы и статистика, 2005, 2007. – 268 с.

2. Каймин В.А. Информатика: учеб.: рек. Мин. обр. РФ / В.А. Каймин. – 5-е изд. – М.: Инфра-М, 2008. – 285 с.

3. Степанов А.Н. Информатика: базовый курс для студентов гуманитар. спец. высш. учеб. заведений / А.Н. Степанов. – 6-е. изд. – СПб.: Питер, 2010. – 720 с.

4. Безручко В.Т. Информатика (курс лекций): учеб. пособие: рек. НМС / В.Т. Безручко. – М.: ФОРУМ: ИНФРА-М, 2006. – 432 с.

III. МЕТОДИЧЕСКИЕ УКАЗАНИЯ (РЕКОМЕНДАЦИИ)

1. Методические указания по изучению дисциплины

Комплексное изучение предлагаемой студентам учебной дисциплины «Информатика» предполагает формирование у студентов теоретических знаний в области информатики, ознакомление с принципами алгоритмизации при решении практических задач, формирование практических навыков по использованию специализированного программного обеспечения.

Процесс по освоению всей совокупности теоретического и практического материала по дисциплине должен быть реализован в течение двух семестров и, проходить в соответствии с предложенным выше планом.

В первом семестре изучение дисциплины «Информатика» основывается на курсе лекций и компьютерном практикуме, включающем освоение студентами программных средств, таких как текстовый редактор Word, пакет презентаций Power Point, владение которыми необходимо любому первокурснику для обучения в вузе.

Во втором и третьем семестрах изучение дисциплины «Информатика» основывается на курсе лекций и компьютерном практикуме, включающем освоение студентами основ программирования и работу с базами данных.

В ходе лекций раскрываются основные теоретические вопросы программы дисциплины, делаются акценты на наиболее сложные и интересные положения изучаемого материала. Это становится возможным благодаря тому, что студенты могут заранее распечатать слайды лекции в качестве основы конспекта (презентация лекции высылается на почтовый ящик студентам за день до лекции), а также за счет применения на лекциях мультимедийных технологий. Материалы лекций являются базовыми для подготовки к экзамену.

Лабораторные занятия проводятся в компьютерных классах с применением специально разработанных учебно-методических пособий, в которых изложены подробные методические рекомендации по изучению каждой темы и выполнению заданий. Наличие таких учебно-методических и дидактических материалов позволяет каждому студенту работать в своем индивидуальном темпе, а также дополнительно прорабатывать изучаемый материал во время самостоятельных занятий, в т.ч. дома. Все эти материалы имеются на кафедре в печатном виде и доступны в электронном виде во всех компьютерных классах.

Вместе с тем, каждая новая тема сначала объясняется преподавателем, рассматривается на примерах, затем для закрепления полученных на занятии знаний студенты выполняют соответствующие упражнения и получают домашние задания. Полученные оценки за выполненные упражнения и домашние задания являются основой для выставления промежуточной и итоговой аттестации. Итоговой аттестацией в первом семестре является зачет, во втором – экзамен. Экзамен проводится по билетам, включающим теоретическую и практическую части, зачет – по тестам.

Для закрепления полученных теоретических и практических знаний студентам в течение всего учебного года предлагаются индивидуальные задания для самостоятельной работы. Особенности выполнения самостоятельной работы и тематика индивидуальных заданий подробно изложены в методических указаниях по их выполнению. Консультирование по выполнению индивидуальных заданий проводится как непосредственно в компьютерных классах (во время консультаций). Контроль выполненных заданий осуществляется либо непосредственно на занятиях, либо на консультациях.

Наличие методических рекомендаций по изучению каждой темы, большого набора заданий для самостоятельной работы по закреплению изучаемого материала (как в виде электронных заданий, так и в виде печатного сборника), компьютерных тестов для контроля знаний по каждой теме позволяет повысить эффективность учебного процесса. Для подготовки к экзамену студентам рекомендуются подготовленные преподавателями кафедры учебник и практикум, включающий терминологическую часть, вопросы для самоконтроля и тесты.

2. Методические указания к лабораторным занятиям

Задачей преподавателя при проведении лабораторных работ является грамотное и доступное разъяснение принципов и правил проведения работ, побуждение студентов к самостоятельной работе, определения места изучаемой дисциплины в дальнейшей профессиональной работе будущего специалиста.

Цель лабораторной работы – научить студентов самостоятельно производить необходимые действия для достижения желаемого результата.

Прежде чем приступить к выполнению лабораторной работы, студенту необходимо ознакомиться с теоретическим материалом, соответствующим данной теме.

Выполнение лабораторной работы целесообразно разделить на несколько этапов:

- формулировка и обоснование цели работы;
- определение теоретического аппарата, применительно к данной теме;
- выполнение заданий;
- анализ результата;
- выводы.

Индивидуальные задания для лабораторных работ должны быть представлены конкретно-практическими и творческими задачами.

На первой ступени изучения темы выполняются конкретно-практические задачи, при решении которых формируется минимальный набор умений. Преподаватель опосредованно руководит познавательной деятельностью студентов, консультирует и подробно разбирает со студентами возникшие затруднения в ходе решения задачи, обращает внимание группы на возможные ошибки.

Вторая ступень изучения темы дифференцируется в зависимости от степени усвоения его обязательного уровня. Студенты, усвоив содержание типовых методов и приемов решения задач, приступают к решению творческих задач. Если уровень знаний и умений, демонстрируемых студентом при контрольном обследовании, не соответствует установленным требованиям, студент вновь возвращается к стандартным упражнениям, но под более пристальным наблюдением преподавателя.

После изучения отдельной темы курса дисциплины, каждый студент получает оценку по результатам выполнения лабораторных работ.

Студенты, пропустившие лабораторные занятия, должны их выполнить во внеаудиторное время и отчитаться до начала зачетно-экзаменационной сессии.

Рекомендации для организации рабочего места студента: для проведения лабораторных работ требуется компьютерный класс с установленным программным обеспечением.

Правила техники безопасности в компьютерном классе:

1. Находиться в компьютерном классе без разрешения преподавателя.
2. Включать без разрешения оборудование.
3. Трогать разъемы соединительных кабелей и проводов.
4. Прикасаться к питающим проводам и устройствам заземления.
5. Включать и выключать аппаратуру без указания преподавателя.
6. Работать в верхней одежде и влажными руками.
7. Класть диски, книги, тетради и другие предметы на монитор и клавиатуру.
8. При появлении запаха гари немедленно прекратите работу, выключите аппаратуру и сообщите об этом преподавателю.

Методические указания к лабораторным занятиям

Лабораторная работа №1. Программирование задач со строковыми данными.

План:

1. Создание текстовых данных.
2. Поиск заданных символов в тексте.
3. Замена символов в тексте.

Объем аудиторных часов: 6 ч.

Объем часов для самостоятельной работы: 5 ч.

Указания к лабораторной работе:

Строкой называется последовательность символов определенной длины. Элементы строки хранятся по 2 в двух байтах памяти ЭВМ.

Переменные типа string могут быть описаны:

Var st1:string[30]; st2:string;

где *st1, st2* – переменные, *string* – строковый тип данных, в квадратных скобках указывается максимальная длина строки (пример 1); если максимальный размер не указан, то он автоматически принимается равным 255, т. к. максимально возможная строка состоит из 255 символов.

Строка имеет определенную длину (не больше некоторого числа), к каждому символу можно обратиться по его номеру (как в массиве), например,

St1[i] – это обращение к *i*-му элементу строки *st1*.

Если ввести больше символов, чем n указанных или возможное максимальное значение, то строка будет равна первым n символам, а остальные будут игнорироваться.

Стандартные функции обработки строковых величин:

1. СКЛЕИВАНИЕ СТРОК.

Под склеиванием понимается последовательное объединение нескольких строк. Для склеивания используется процедура **Concat(str1,str2,…,strn)** – результат – строка, образованная склеиванием строк *str1, str2, ..., strn*.

Пример. Str1:=’object’;
Str2:=’pascal’;
Str3:=Concat(str1,’ ‘,str2);

Значение str3=object pascal.

Функция Pos аналогична операции ‘+’.

Пример. Var str1,str2,str3:string[10];
begin
Str1:=’object’;
Str2:=’pascal’;
Str3:=str1+’ ‘+str2;

Значение str3=object pas.

Итоговая строка может состоять максимум из 10 символов. Если в строке будет стоять больше 10 символов, то будут взяты только первые 10, а остальные будут игнорироваться (см. пример).

2. ДЛИНА СТРОКИ.

Под длиной строки понимается количество введенных символов. Но она не может превышать максимально возможной длины. Это значение можно определять при помощи функции **Length(str)**, результат которой целое число, равное количеству символов.

Пример. str1:='март';
str2:='мама мыла раму';
k1:=Length(str1);
k2:=Length(str2);

В результате значения целых переменных будут равны k1=4, k2=15.

3. ПОИСК ПОДСТРОКИ В СТРОКЕ.

Для этого используется функция **Pos(str1,str)** – позволяет определить положение подстроки *str1* в строке *str*. Результат – целое число, равное номеру позиции, с которой в строке *str* начинается строка *str1*. Если подстрока str1 не найдена, то значение функции равно 0.

Пример. Str1:=’cal’;
Str:=’object pascal’;
Str2:=Pos(str1,str);

Значение str2=11.

4. КОПИРОВАНИЕ.

Для этого используется функция **Copy(Str,n,m)** - копирует m символов строки *str*, начиная с *n* - го символа; при этом исходная строка не меняется. Можно результат этой функ-

ции присваивать другой строке или сразу выводить на экран.

Пример. str1:='ABCDEFGH';
str2:='abcdefgh';
str3:=Copy(str1,4,3);

Значение переменной str3='DEF'.

Стандартные процедуры обработки строковых величин:

1. УДАЛЕНИЕ

Для этого используется процедура **Delete(str,n,m)**, которая вырезает из строки *str* m символов, начиная с *n* – го; таким образом, строка изменяется.

Пример. Str1:='ABCDEFGH';
Delete(str1,3,4);

После выполнения этих операторов из строки будут удалены 4 символа, начиная с 3-его, т. е. Str1='ABGH'.

2. ВСТАВКА.

Для этого используется процедура **Insert(str1,str2,n)** – вставка строки *str1* в строку *str2*, начиная с *n* – го; при этом строка *str1* остается без изменения, а строка *str2* получает новое значение.

Пример. Str1:='ABCDEFGH';
Str2:='abcdefgh';
Insert(str1,str2,3);

После выполнения этих операторов строка *str2* будет иметь вид: Str2='abABCDEFGHcdefgh'.

Такой же результат будет после выполнения следующей последовательности операторов:

str2:='abcdefgh';
Insert('ABCDEFGH',str2,3);

Пример 1. Дана строка символов. Удалить из строки заданный символ.

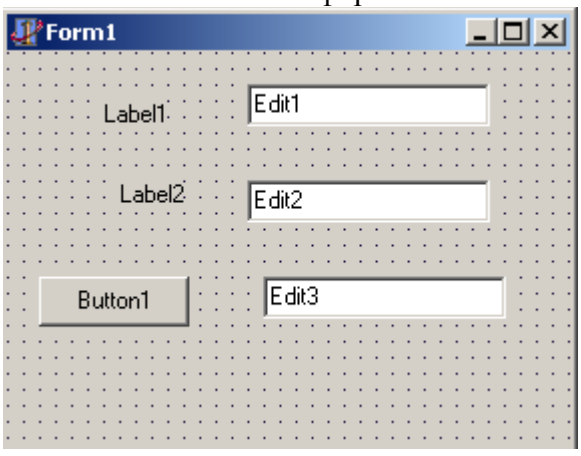
Определим переменные задачи:

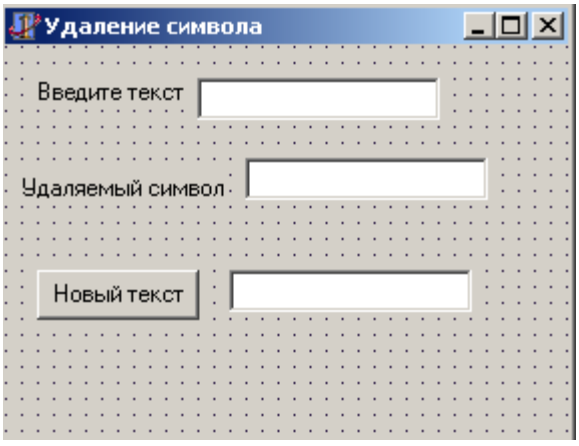
Входные данные: str: строкового типа – вводимый текст; c: строкового типа – удаляемый элемент.

Промежуточные переменные: i: целого типа - параметр элемента, номер элемента строки, изменяющийся с шагом 1;

Результат: str: строкового типа – преобразованный текст.

Решение данной задачи рассмотрим для цикла с постусловием. Расположим следующие компоненты на окне формы.





Свойства выбранных компонент:

- Form1 – Name – Form1
  Caption – Удаление символа
- Label1 – Name - Label1
  Caption – Введите текст
- Label2 – Name - Label2
  Caption – Удаляемый символ
- Button1 – Name - Button1
  Caption – Новый текст

– Edit1 – Text – пусто
– Edit2 – Text – пусто
– Edit3 – Text - пусто

Программа решения задачи с помощью цикла с постусловием следующие:

```
procedure TForm1.Button1Click(Sender: TObject);
var
 str,c:string;
 i:integer;
begin
str:=edit1.text;
c:=edit2.text;
i:=0;
repeat
 if str[i]=c
  then delete(str,i,1)
   else i:=i+1;
until i>length(str);
edit3.text:=str;
end;
```

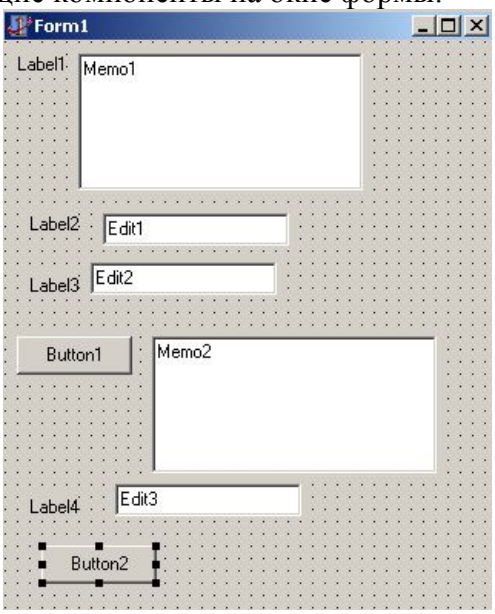
Пример 2. Составить программу, заменяющую в тексте данный слог на введенный, и подсчитывающую количество замен.

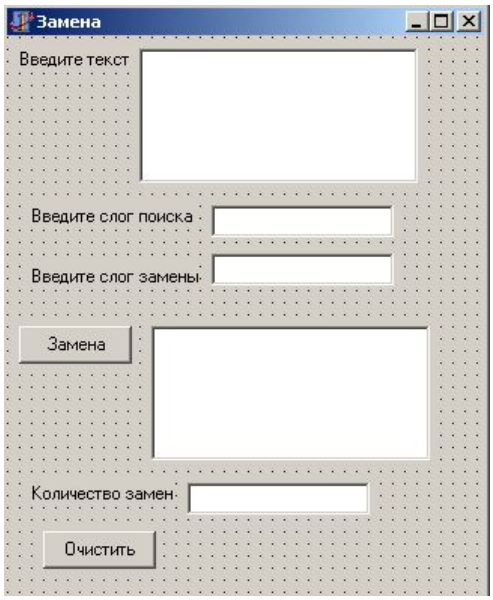
В задаче необходимо ввести текст, искомый слог и слог замены. Для этого будем использовать входные переменные: S - текст; a - искомый слог; b - слог замены.

Результатом будет являться: новый текст (уже заданная переменная S); k - количество замен.

Промежуточные переменные будут описываться в ходе написания программы.

Решение данной задачи рассмотрим для цикла с предусловием. Расположим следующие компоненты на окне формы.





Свойства выбранных компонент:

– Form1 – Name – Form1
Caption - Замена
– Label1 – Name - Label1
Caption – Введите текст
– Label2 – Name - Label2
Caption – Введите слог поиска
– Label3 – Name - Label3
Caption – Введите слог замены
– Label4 – Name - Label4
Caption – Произведено замен
– Button1 – Name - Button1
Caption – Замена

– Memo1 – Name – Memo1 Lines – пусто
– Memo2 – Name – Memo2 Lines – пусто
– Edit1 – Text – пусто
– Edit2 – Text – пусто
– Edit3 – Text - пусто

В рассматриваемой программе будет происходить два события: замена и очистка.

```
procedure TForm1.Button1Click();
var Str,a,b:string;
    i,k:integer;
begin
Str:=memo1.text;
a:=edit1.text;
b:=edit2.text;
k:=0;
i:=Pos(a,str);
while i<>0 do
 begin
  delete(Str,i,length(a));
  Insert (b,Str,i);
  k:=k+1;
  i:=Pos(a,str);
 end;
memo2.text:=Str;
Edit3.text:=IntToStr(k);
Memo2.visible:=true;
Edit3.visible:=true;
label4.visible:=true;
end;
```

Обратите внимание! При присваивании:

текстовой компоненте значения строковой переменной;

строковой компоненте значения компоненты функции перевода не используются. Например, Memo2.text:=S; S:=Memo1.text.

Лабораторная работа №2 Программы построения графиков.

План:

1. Компонент построения графиков TChart.
2. Графическое представление данных.

Объем аудиторных часов: 4 ч.

Объем часов для самостоятельной работы: 5 ч.

Указания к лабораторной работе:

Существуют классы-надстройки, созданные для использования в графических инструментах Windows: для контекста – класс TCanvas, для шрифта – TFont, для пера – TPen, для кисти – TBrush.

Класс TFont

С помощью класса TFont создаются объект-шрифт для любого графического устройства (в частности, объекты на экране).

Для данного класса характерны свойства:

Color: TColor;	Цвет шрифта.
Height: Integer;	Высота шрифта в пикселях экрана.
Name: TfontName;	Имя шрифта.
Size: Integer;	Высота шрифта в пунктах (1/72 дюйма).
Style = [fsbold, fsItalic, fsUnderLine, fsStri-	Стиль шрифта. Может принимать комбинацию элементов: fsbold (жирный), fsItalic (курсив), fsUnderLine (подчеркнутый), fsStri-

keOut]	keOut (перечеркнутый).

Почти каждый компонент имеет свойство Font:TFont. Слева от свойства стоит знак «+», который указывает, что свойство вложенное. Щелчок мышью на знаке плюс открывает вложенные свойства класса Tfont.

Классы TPen, TBrush.

С помощью класса TPen (Перо) создается объект Перо, служащий для вычерчивания линий, контуров графических рисунков.

Свойства класса:

Color: TColor;	Цвет вычерчиваемых пером линий.
Style: TPenStyle;	Определяет стиль линий: сплошная, пунктирная и т.д.
Width: Integer;	Толщина линий в пикселях экрана.
Mode: TpenMode;	Определяет способ взаимодействия линий с фоном.

Объект класса TBrush (Кисть) служит для заполнения внутреннего пространства замкнутых фигур.

Свойства класса:

Bitmap: TBitMap;	Содержит растовое изображение, которое будет использоваться кистью для заполнения. Если это свойство определено, то свойства Color, Style игнорируются.
Color: TColor;	Цвет вычерчиваемых пером линий.
Style: TBrushStyle;	Определяет стиль линий заполнения: сплошная закраска, диагональные линии, горизонтальные линии и т.д.

Класс TCanvas

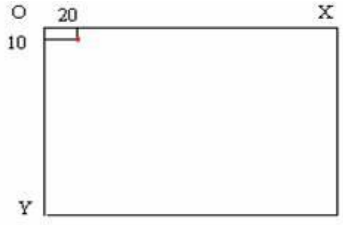
Данный класс создает «канву», на которой можно рисовать. Объекты класса TCanvas автоматически создаются для всех видимых неоконых компонентов.

Класс имеет свойства Pen: TPen, Brush: TBrush, Font: TFont.

Задание цвета пера и кисти можно осуществить несколькими способами. Например, непосредственно указав стандартные цвета Windows (clred, clblue, clgreen и т.д.). Наиболее распространенным способом задания цвета в Windows является цветовая модель RGB (по первым буквам названий цветов: Red – красный, Green – зеленный, Blue – голубой). Комбинируя эти три цвета, можно получить любой цвет. Для этого интенсивность каждого из трех составляющих цветов задается значением от 0 до 255.

Рассмотрим основные свойства и методы класса TCanvas.

Pixels[x,y: Integer]: TColor – это двумерный массив, который отвечает за цвет пикселя. Координаты пикселя в массиве определяются согласно рисунка.



Например, для размещения красной точки в объекте Label1 с координатами (10,20) следуют записать:

Label1.Canvas.Pixil(10,20):=clred;

PenPos:Tpoint – определяет текущее положение пера в пикселях относительно левого верхнего угла канвы.

MoveTo(X,Y: integer) – перемещение пера, курсор устанавливается, не рисуя, в позицию (Х,У).

LineTo(X,Y) – рисуется отрезок от позиции, в которой установлено перо, до позиции (Х,У).

PolyLine(Points: array of TPoints) – рисование контура многоугольника. Для этого используется массив POINT:TPOINT, передающий последовательность точек, которые функция

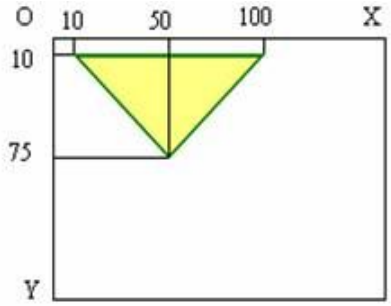
PolyLine соединяет линиями. Чтобы многоугольник получился замкнутым, необходимо соединить позиции последней и первой точек. Пример рисования треугольника:

label1.Canvas.Pen.Color:=clgreen;

Label1.Canvas.PolyLine (Points(10,10), Points(50,75), Points(100,10), Points(10,10));

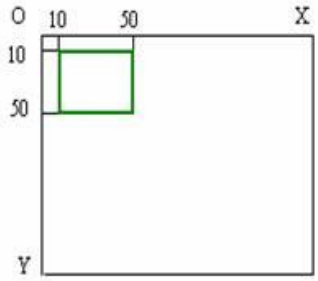
Polygon (Points: array of TPoints) – рисование закрашенных многоугольников. Контур рисуется установленными свойствами пера, при закрашивании используются текущие свойства кисти. Пример рисования закрашенного треугольника:

```
Label1.Canvas.Pen.Color := clgreen;
Label1.Canvas.Pen.Width := 4;
label1.Canvas.Brush.Color:=clYellow;
Label1.Canvas.Polygon (Points(10,10),
Points(50,75), Points(100,10),
Points(10,10));
```



Rectangle(x1,y1,x2,y2: integer) – рисование контура прямоугольника. Рисуется контур прямоугольника, где (x1,y1) – координаты верхнего левого угла, (x2,y2) – координаты нижнего правого угла. При рисовании используются текущие свойства пера, которые можно задать перед применением процедуры рисования. Например:

```
Label1.Canvas.Pen.Color := clgreen;
Label1.Canvas.Pen.Width := 4;
Label1.Canvas.Rectangle(10,10,50.50);
```

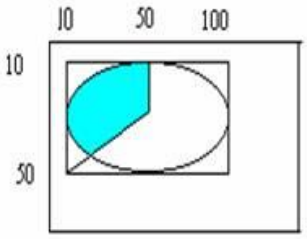


FillRect(Rect(x1,y1,x2,y2:integer)) – рисуется закрашенный прямоугольник, где функция Rect(x1,y1,x2,y2) возвращает размер прямоугольной области аналогично функции Rectangle, в которой (x1,y1) – координаты верхнего левого угла, (x2,y2) – координаты нижнего правого угла. Контур рисуется установленными свойствами пера, при закрашивании используются текущие свойства кисти.

Ellipse(X1,Y1,X2,Y2: integer) – рисование закрашенных окружностей и эллипсов. Чертит эллипс в охватывающей прямоугольной области. При рисовании окружностей указываются размеры квадратной области.

Pie(x1,y1,x2,y2,x3,y3,x4,y4:integer) – рисование секторов эллипса. При этом первые четыре параметра указывают размер прямоугольной области. Параметры X3,У3,X4,У4 определяют часть эллипса, которая будет показана. Начало дуги лежит на пересечении эллипса и луча, проведенного из его центра в точку (x3,y3), а конец – на пересечении с лучом из центра в точку (x4,y4). Дуга чертится против часовой стрелки. Например:

```
Label1.Brush.Color := clblue;
Label1.Canvas.Pie (10,10,100,50,50,10,10,50);
```



TextOut(x,y:integer; S:string) – выводит текстовую строку S, начиная с позиции (x,y). Перед применением функции можно задать параметры шрифта. Например:

Label1.canvas.font.Style:=[fsbold];

Label1.canvas.TextOut (1,1,'чертеж');

Компонент TShape, TImage, TPaintBox, TChart.

 TShape – стандартная фигура. Компонент рисует одну из простейших геометрических фигур: прямоугольник, квадрат, скругленный прямоугольник, скругленный квадрат, эллипс, окружность.

Свойства компонента:

Shape = (stRectangle, stSquare, stRoundRect, stRoundSquare, stEllipse, stCircle);	Определяет вид фигуры: stRectangle – прямоугольник, stCircle – окружность, stSquare – квадрат, stEllipse – эллипс, stRoundRect – скругленный прямоугольник, stRoundSquare – скругленный квадрат.
+ Brush: Tbrush;	Кисть, служит для заполнения внутреннего пространства фигур. Имеет вложенные свойства (см. пункт 14.2).
+ Pen: Tpen;	Перо, служит для вычерчивания линий, контуров графических рисунков.

 TImage – отображение картинок. Этот компонент служит для размещения на форме трех видов изображения: растровой картинки, пиктограммы или метафайла. Изображение содержится в свойстве компонента *Picture*. Свойство *Canvas* позволяет отредактировать растровое изображение.

TPaintBox – окно для рисования. Компонент размещает на форме прямоугольную область (канву) для рисования. Если при рисовании фигуры ее размеры окажутся больше компоненты TPaintBox, то фигура окажется усеченной – не выйдет за пределы области рисования. Создав обработчик события OnPaint для TPaintBox, можно передвигать изображение по форме, изменяя у TPaintBox свойства Left и Top. TPaintBox использует свойство ALIGH для удержания изображения сверху, снизу, сверху, внизу формы или заполнять всю область формы.

 TChart – построитель графиков. Компонент предназначен для графического представления числовых данных.

После размещения компонента на форме нужно щелкнуть по нему правой кнопкой мыши для вызова контекстного меню и выбрать команду *Edit Chart,* после чего появляется окно редактирования компонента.

В окне редактирования нужно нажать кнопку *Add* и выбрать подходящий тип графика. Каждый выбранный график называется серией, которые нумеруются, начиная с нуля. Доступ к сериям осуществляется с помощью закладки *Series*.Окно редактирования имеет закладки для задания заголовка графика Title, редактирования осей – Axis.

После закрытия окна редактора компонент будет содержать примерный вид графика. Однако его реальный вид зависит от фактических данных, которые создаются в работающей программе. В программе обращаются к свойству *SeriesList[<номер серии>]* – список серий; первая созданная серия имеет индекс 0, вторая 1 и т.д. Свойству *SeriesList* доступны методы: *AddX* – добавить данные по X, *AddY* – добавить данные по Y, *AddXY* – добавить данные по X и Y.

Например, обработчик события OnCreate формы создает график y=5-sin(x) (график будет показан при открытии окна программы):

```
procedure TForm1.FormCreate(Sender: TObject);
const PI=3.14;
var x:integer; y:real;
begin
  for x:= -100 to 100 do begin
                          y:= 5-sin(x/pi);
                         Chart1.SeriesList[0].AddXY(x,y);
                          end;
end;
```

Задания

1. Телевизор Samsung 34KL стоит x1, x2, x3 руб. соответственно в магазине №1, №2, №3. В каждом магазине при покупке TV дается скидка r1%, r2%, r3% соответственно. Вычислить реальную стоимость покупки в каждом магазине, отобразите расчеты в виде гистограммы.

2. Известен средний ежеквартальный расход электроэнергии P кВ/час. При этом извест-

но: в 1 квартале расход на 15% выше среднего, во 2 и 4 квартах – на 5% выше среднего, в 3 квартале - на 8% ниже среднего. Вывести расход электроэнергии в каждом квартале p1, p2, p3, p4. Изобразите ежеквартальный расход электроэнергии в виде круговой диаграммы.

3. Для матрицы объема сбора урожая B (1-4 кв., 2001-2004гг.) найти среднеквартальное значение C2001, C2002, C2003, C2004 каждого года. Изобразить круговую диаграмму данных среднеквартальных значений.

Лабораторная работа №3 Страница Диалоги.

План:

1. Знакомство со стандартными диалогами Windows.
2. Этапы работы со стандартными диалоговыми окнами.

Объем аудиторных часов: 4 ч.

Объем часов для самостоятельной работы: 5 ч.

Указания к лабораторной работе:

На странице Dialogs представлены компоненты для вызова стандартных диалогов Windows: TOpenDialog – открыть файл, TSaveDialog - сохранить файл, TFontDialog - настройка шрифта, TColorDialog - выбор цвета, TPrintDialog – печать, TPrinterSetupDialog - настройки принтера, TFindDialog - поиск строки, TReplaceDialog - поиск с заменой.

Работа со стандартными диалоговыми окнами осуществляется в 3 этапа:

1. На форму помещается соответствующий компонент и осуществляется настройка его свойств. Сам компонент не виден в момент работы программы (невизуальный компонент), видно лишь создаваемое им стандартное окно.

2. Осуществляется вызов стандартного для диалогов метода *Execute*, который создает и показывает на экране диалоговое окно. **Execute** – логическая функция, возвращает в программу True, если результат диалога с пользователем был успешным.

3. Анализ результата метода Execute, использование введенных с помощью диалогового окна данных – имени файла, настроек принтера, выбранного шрифта и т. д.

Основные компоненты страницы Dialogs:

1. TColorDialog.

Компонент создает и обслуживает стандартное окно выбора цвета. Свойство *Color* содержит стандартный цвет в окне диалога.

Пример использования диалога в программе может быть следующим:

If ColorDialog1.Execute (вызов окна диалога)

Then Memo1.Color:=ColorDialog1.Color;(memo1 залить цветом, выбранным в окне диалога «Цвет»)

2. TFontDialog.

Компонент создает и обслуживает стандартное окно выбора шрифта. Свойство компонента *Font* содержит выбранные параметры шрифта в окне диалога.

Пример использования диалога в программе может быть следующим:

If FontDialog1.Execute (вызов окна диалога)

Then Memo1.Font:=FontDialog1.Font;(задать стили шрифта в memo1, выбранных в окне диалога «Шрифт»)

3. TOpenDialog и TSaveDialog.

Диалоги открытия и сохранения файла.

При открытии или сохранении файла пользователь в окне диалога указывает путь и имя файла, которое возвращается свойством *FileName:string*.

При открытии файла пользователь может ввести произвольное имя и, следовательно, указать несуществующий файл. Чтобы этого избежать, можно проверить существующий файл функцией *FileExistls:boolean*.

Свойство *Filter:string* используется для фильтрации (отбора) файлов, показываемых в диалоговом окне. Это свойство можно устанавливать в программе, например,

OpenDialog.Filter:=’текстовые файлы/*.txt/файлыObject Pascal/*.pas’;

(задает две маски отбора файлов с расширением .txt и .pas)

или в окне Инспектора Объектов при выборе свойства *Filter*, открывающего окно специального редактора.

Свойство *Title:string* задает заголовок диалогового окна.

Установить начальный каталог окна диалога можно с помощью свойства *InitialDir: string* в программе или в окне Инспектора Объектов.

В свойстве *DefaultExt*:string[3] указывается расширение, присваиваемое сохраняемому файлу по умолчанию (если пользователь не указал другое).

Для некоторых компонент характерны методы:

– *SavetoFile*(имя файла:string) – сохраняет информацию в виде файла с указанным именем;

– *LoadFromFile(*имя файла:string*)* – ститывает содержимое файла.

Пример использования компоненты OpenDialog для чтения информации с файла и записи ее в Memo1.

```
Begin
OpenDialod1.InitialDir:='C:\Мои документы';(устанавливаем начальный каталог в окне диалога «Открытие файла»)
OpenDialog1.Filter:='текстовые файлы|*.txt|файлы Object Pascal|*.pas'; (настраиваем диалог на отбор текстовых файлов)
If OpenDialog1.Execute and FileExists(OpenDialog1.FileName) (выполняем диалог и проверяем существование выбранного диалогом файла)
  Then Memo1.Lines.LoadFromFile(OpenDialog1.FileName); (считываем содержимое файла в Memo1)
End;
```

Пример использования компоненты SaveDialog для записи содержимого компоненты Memo1 в файл.

```
Begin
SaveDialod1.InitialDir:='C:\Мои документы';
SaveDialog1.Filter:='текстовые файлы|*.txt|файлы Object Pascal|*.pas';
SaveDialog1.DefaultExt:='txt';(указываем расширение файла по умолчанию)
If SaveDialog1.Execute
  Then Memo1.Lines.SaveToFile(SaveDialog1.FileName); (записываем содержимое Memo1 в файл)
End;
```

4. TOpenPictureDialog и TSavePictureDialog – диалоги открытия и сохранения изображений.

Специализированные диалоги для открытия и сохранения графических файлов отличаются от TOpenDialog и TSaveDialog двумя обстоятельствами:

– В них предусмотрены стандартные фильтры для выбора графических файлов (с расширением .bmp, .ico, .wmt, .emf и др.), т. е. Отсутствует свойство Filter.

– В окна диалога включена панель для предварительного просмотра выбираемого файла.

Считывать содержимое графического файла можно только в компоненты, работающие с графикой, рисунками (Image и др.).

Вызов диалогов происходит программно, например:

```
OpenPictureDialog1.InitialDir:='C:\Мои документы';
If OpenPictureDialog1.Execute and FileExists(OpenPictureDialog1.FileName)
  Then Image1.Picture.LoadFromFile(OpenPictureDialog1.FileName);
```

Дополнительные сведения:

Image(Additional) – изображение картинок. Изображения содержатся в свойстве *Picture*.

MainMenu – горизонтальное меню. Свойство *Items* позволяет определить пункты меню.

Задание

1. Разместить компоненты Memo(текстовая область) Image(графика).
2. Создать горизонтальное меню с пунктами:

– Текст: цвет, шрифт, открыть, сохранить (для работы с текстовой областью)

– Графика: открыть (для размещения файла с графическими рисунками)

– Окно: выход, о программе (ShowMessege).

3. Написать код для каждого пункта Memo (реакцию на событие OnClick).

Не забыть разместить на форме соответствующие компоненты.

Лабораторная работа №4. Базы данных.

План:

1. Создание таблиц различными способами.
2. Схема данных.
3. Работа с запросами.
4. Создание форм и отчетов.

Объем аудиторных часов: 4 ч.

Объем часов для самостоятельной работы: 5 ч.

Указания к лабораторной работе:

Задание 1. Запустить MS Access.

Задание 2. Создать новую базу данных и сохранить ее с именем TYR в своей папке.

Задание 3. В ходе выполнения лабораторной работы вам необходимо решить следующую задачу:

Разработать информационную модель туристического предприятия, включающие следующие реквизиты: код тура, наименование тура, продолжительность тура, цена тура, код страны, название страны, виза (нужна или нет), валюта страны, код менеджера, Фамилия Имя Отчество, телефон. Основные условия:

в одну страну может быть несколько туров, но каждый тур предусматривает посещение только одной страны;

один менеджер курирует несколько туров, но каждый тур имеет только одного менеджера-куратора.

Для создания таблиц в среде MS Access необходимо открыть диалоговое окно базы данных в режиме таблицы. Далее: если нажать на кнопку *Создать*, на экране появится диалоговое окно, в котором будет предложено несколько способов работы с таблицей.

Мастер таблиц

Для создания таблицы в режиме мастера необходимо:

1. Выбрать компонент *Таблицы*.
2. Выбрать способ *Создание таблицы с помощью мастера*.
3. Нажать кнопку *Создать*.
4. В диалоговом окне *Новая таблица* выбрать *Мастер таблиц* и нажать кнопку *ОК*.

Далее создание таблицы выполняется по шагам.

Первый шаг. В списке слева перечислены образцы таблиц. Под списком расположены кнопки *Деловое применение* и *Личное применение*. Содержание списка зависит от вашего выбора. При выборе таблицы в списке в середине появится перечень предлагаемых полей. Для перемещения нужного поля из среднего списка в список слева щелкните на имени поля и затем на кнопке  . Если вам не нравится какое-либо название, его можно изменить, выделив поле и щелкнув на кнопке *Переименовать*.

Задание 4. Выбрать кнопку *Деловое применение* в списке образцы таблиц *Контакты*.

Задание 5. Поместить в список *Поля новой таблицы* поля из списка *Образцы полей*: Код страны (выбрать поле Код контакта и переименовать его); Название (выбрать поле Страна/регион и переименовать его); Виза (выбрать поле Код ТипаКонтакта и переименовать его); Валюта (выбрать поле Код ТипаКонтакта и переименовать его) и нажать кнопку *Далее*.

Второй шаг. Задается имя таблицы и определяется ключевое поле в новой таблице.

Если выбран режим автоматического определения ключа и в новую таблицу включено поле-счетчик, то последнее выбирается в качестве ключевого. Иначе программа автоматически создает еще одно поле в таблице в качестве ключевого. Второй путь – задание поля самостоятельно.

Задание 6. Задать имя таблицы *Страна* и установить самостоятельное определение ключа. Нажать кнопку *Далее*.

Задание 7. Определить ключевое поле *Код страны*.

Третий шаг. Предоставляется возможность связать новую таблицу с другими таблицами БД. Список существующих таблиц БД выводится в окне. Если в какой-либо из таблиц есть поле, совпадающее с ключом создаваемой таблицы, Access предложит наличие связи. С выбором MS

Access можно согласиться, отказаться или создать вручную.

Четвертый шаг. Определение режима, который активизирован после завершения работы *Мастера*.

Переключатель *Изменить структуру таблицы* означает переход в режим конструктора для новой таблицы. Он выбирается, если необходимо доделать то, что не смог выполнить *Мастер таблиц*: ввести новые поля, придать им необходимые свойства, переопределить ключ и т.д. Переключатель *Ввести данные непосредственно в таблицу* приведет к тому, что таблица будет открыта для просмотра в табличной форме. Переключатель *Ввести данные в таблицу с помощью формы, создаваемой Мастером*, заставит создать форму для новой таблицы.

Задание 8. Выбрать переключатель *Ввести данные непосредственно в таблицу* и нажать кнопку *Готово.*

Режим конструктора

В режиме конструктора можно не только вводить имена полей, но также выбирать их тип и задавать их свойства.

Задание 9. Выбрать таблицу *Страна* и нажать кнопку *Конструктор*. Появится окно конструктора таблицы.

В открывшемся окне конструктора необходимо указывать *Имя поля* и *Тип данных*, это нужно для создания имен и значений полей для дальнейшей работы (поле – это свойство рассматриваемого объекта, полями являются столбцы нашей таблицы).

Задание 10. Задать для всех полей таблицы *Страна Тип данных – Текстовой*. Закрыть окно конструктора с сохранением изменений.

Задание 11. Открыть таблицу *Страна* и заполнить ее.

Режим таблицы

Режим таблицы – это превосходный способ создания простых таблиц, подходящих для ситуации, когда вам требуется немедленно заполнить их. Создание таблицы заключается в задании полям имен и вводе данных. Для определения имени поля нужно дважды щелкнуть на *Поле1* или других именах полей или щелкнуть правой кнопкой мыши и выбрать команду *Переименовать столбец*. После этого можно вводить данные в таблицу. Столбцы таблицы можно изменить в размерах, удалять, скрывать.

Задание 12. В режиме таблицы создайте таблицу *Менеджер*, в которой содержится информация о менеджерах турфирмы.

При сохранении таблицы в режиме таблицы программа выдаст запрос о задании ключевого поля.

На вопрос программы необходимо ответить *Нет* и задать ключевое поле самостоятельно. Для этого нужно открыть таблицу в режиме конструктора, установить курсор напротив ключевого поля и нажать кнопку  на панели инструментов.

Задание 13. Для таблицы *Менеджер* в режиме конструктора установить ключевое поле *Код менеджера* и сохранить макет таблицы.

Задание 14. Создать таблицу *Тур* в режиме конструктора, имеющую следующие поля:

Код тура (тип данных – текстовой, размер поля – 10 символов).

Название тура (тип данных – текстовой, размер поля – 20 символов).

Продолжительность тура (тип данных – числовой, размер поля – длинное целое).

Цена (тип данных – денежный, формат поля – евро).

Код менеджера (тип данных – Мастер подстановок, размер поля – 15 символов).

Код страны (тип данных – Мастер подстановок, размер поля – 10 символов).

Задание 15. Выбрать в качестве ключевого поле *Код тура*. Не заполнять таблицу *Тур* без установления схемы данных.

Схема данных

После создания таблиц, содержащих данные, относящиеся к различным аспектам базы данных, разработчик должен продумать, каким образом MS Access будет объединять эти данные при их извлечении из базы данных. Первым шагом при этом является определение связей между таблицами.

Чтобы созданные таблицы работали как единое целое, между ними необходимо установить связь. Связь между таблицами устанавливает тип отношений между совпадающими значениями в ключевых полях – обычно между полями разных таблиц, имеющими одинаковые имена. В большинстве случаев с ключевым полем одной таблицы, являющимся уникальным идентификатором каждой записи, связывается внешний ключ другой таблицы. Связь между таблицами может быть трех типов: *отношение* «*один-ко-многим*» (1 – ∞), *отношение* «*многие-ко-многим*» (∞ – ∞), *отношение* «*один-к-одному*» (1 – 1).

Для установки связи между таблицами необходимо выбрать команду *Сервис* → *Схема данных* или нажать кнопку  на панели инструментов. Если для данной базы данных впервые открывают окно *Схема данных* или эти отношения еще не сохранялись в предыдущих использованиях, появляется новое окно *Схема данных* вместе с диалоговым окном *Добавление таблицы*.

Если существует ранее сохраненная схема данных, эта схема заполняет окно и диалоговое окно *Добавление таблицы* не появляется. Окно *Схема данных* отображает таблицы и отношения, которые были определены между этими таблицами. Отношения показываются сплошными линиями, соединяющими ключевые поля таблиц схемы данных. Чтобы добавить таблицу в схему данных, либо дважды щелкните на ее имени в диалоговом окне *Добавление таблицы*, либо выберите ее имя в списке и щелкните на кнопке *Добавить*. Это действие размещает таблицу в схеме и показывает все отношения, которые уже были установлены в ней.

Задание 16. Заполнить окно *Схема данных*.

Связь между полями устанавливают путем перетаскивания имени поля из одной таблицы в другую на соответствующее ему связанное поле. После перетаскивания открывается диалоговое окно *Изменение связей*, в котором можно задать свойства образующейся связи.

Включение флажка *Обеспечение условия целостности данных* позволяет защититься от случаев удаления записей из одной таблицы, при которых связанные с ними данные других таблиц останутся без связи. Чтобы условие целостности могло существовать, поле основной таблицы должно обязательно быть ключевым и оба поля должны иметь одинаковый тип. Флажки *Каскадное обновление связанных полей* и *Каскадное удаление связанных записей* обеспечивают одновременное обновление или удаление данных во всех подчиненных таблицах при их изменении в главной таблице.

Задание 17. Установить связи между таблицами *Страна* и *Тур*.

Задание 18. Установить связи между таблицами *Тур* и *Менеджер*.

Задание 19. Сохранить установленные связи между таблицами.

Задание 20. Заполнить таблицу *Тур*.

Тур : таблица

Код тура	Название тура	Продолжительность тура	Цена	Код менеджер	Код страны
CYP-1	Отдых	14	810,00€	M-2	CYP
CYP-2	Обучение	14	10 000,00€	M-2	CYP
CYP-3	Отдых на море	7	670,00€	M-2	CYP
FR-1	Париж-Париж	7	780,00€	M-4	FR
FR-2	Рождество	7	920,00€	M-4	FR
FR-3	Обучение	10	1 800,00€	M-4	FR
RUS-1	Сочи	15	400,00€	M-4	RUS
RUS-2	Золотое кольцо	14	950,00€	M-3	RUS
RUS-3	Сибирь	25	300,00€	M-3	RUS
UK-1	Обучение	15	2 100,00€	M-3	UK
UK-2	Экскурсия в Лондон	7	690,00€	M-1	UK
USA-1	Диснейленд	5	1 200,00€	M-1	USA
USA-2	Долина смерти	5	900,00€	M-1	USA
USA-3	Отдых	14	3 000,00€	M-1	USA
USA-4	Обучение	10	1 500,00€	M-1	USA

Запись: 16 из 16

Задание 21. Отредактировать структуру базы данных:

в базу данных включить информацию об иностранном языке, который владеет каждый менеджер, и название столиц государств;

в базу данных включить информацию об авиакомпаниях, с которыми сотрудничает

фирма: код авиакомпании, название авиакомпании, Фамилия Имя Отчество конкретного лица, телефон. Увязать эти данные с исходной базой данных, учитывая, что каждый тур обслуживается одной авиакомпанией и одна авиакомпания может обслуживать несколько туров, например, в одну и ту же страну.

Задание 22. Изменить код одного из менеджеров. Проверить изменение кода в других таблицах.

Создание запросов

С помощью запросов можно просматривать, анализировать и изменять данные из нескольких таблиц. Они также используются в качестве источника данных для форм и отчетов.

Создание запроса-выборки

Для создания запроса необходимо выбрать объект *Запросы*.

Далее необходимо выбрать режим *Создание запроса с помощью мастера* и нажать кнопку *Создать*. В появившемся диалоговом окне *Новый запрос* выбрать *Простой запрос*. Затем указывается имя таблицы, по которой осуществляется поиск, и выбираются требуемые поля. Далее указывается имя запроса. После нажатия кнопки *Готово* на экран выводится результат запроса.

Задание 23. Создать запрос по таблице *Менеджер*, выводящий список всех менеджеров турфирмы с указанием их телефонов.

Создание запроса на выборку с помощью конструктора

При составлении запросов используются операторы:

<, >, <=, >=, <> – для задания сравнения;

In (значенте1, значение2, …) – для проверки включения значения в список;

Between…and – для проверки вхождения в интервал значений;

And, or, not – для проверки условия;

Like – вводимое значение сравнивается с образцом;

? – заменяет один любой символ;

* – заменяет произвольное количество любых символов;

- заменяет любую цифру;

! после первой скобки – для поиска символа, который не входит в указанный набор символов.

Для создания запроса необходимо выбрать объект *Запросы*. Далее выбирается режим *Создание запроса в режиме конструктора* и нажимается кнопка *Создать*. В появившемся диалоговом окне *Новый запрос* выбирается *Конструктор*. На экране появляется окно запроса.

Верхняя часть окна запроса, называемая панелью таблицы, показывает отношение между запрашиваемыми таблицами.

Нижняя часть, называемая сеткой запроса или сеткой QBE (query by example – запрос по образцу), показывает поля из таблиц и условия отбора, по которому они были запрошены. Она имеет следующие строки:

Поле. Перечислены используемые в запросе поля. Поля размещаются по столбцам слева направо.

Имя таблицы. Отображается имя таблицы, из какой выбрано поле.

Сортировка. Позволяет упорядочивать записи в результирующей таблице.

Вывод на экран. Отменяется показ на экране того или иного поля (по умолчанию все поля, участвующие в запросе, выводятся на экран).

Условие отбора. Вводится критерий поиска записей.

При создании нового запроса все таблицы или запросы перечисляются в диалоговом окне *Добавление таблицы*. Выделив желаемую таблицу и/или запрос и щелкнув на кнопке *Добавить*, эти элементы располагают в панели таблиц окна запроса. Иначе, этого можно достичь, дважды щелкнув на каждом элементе в диалоговом окне *Добавление таблицы.*

Для примера предположим, что в имеющейся базе данных *TYR* необходимо найти все туры, цены которых меньше 800 евро, и на экран вывести название страны, название ту-

ра, его продолжительность и цену.

В окне *Добавление таблицы* следует выделить таблицы *Тур* и *Страна* и добавить их в поле конструктора. Связи между таблицами появляются автоматически в соответствии со схемой базы данных. Затем в строке *Поле* надо последовательно указать те поля, которые будут использованы в запросе. В строке *Условие отбора* в соответствующем поле *Цена* указать критерий отбора записей – <800.

Для вывода информации согласно условию запроса необходимо нажать кнопку  на панели инструментов.

Задание 24. Создать запрос, содержащий:

названия туров, продолжительность которых составляет от 7 до 20 дней;

названия стран, денежные единицы которых начинаются на букву «Ф»;

фамилию, имя, отчество менеджеров, владеющих определенным иностранным языком (список должен быть отсортирован по возрастанию).

Создание запроса с параметром

Запросы с параметрами целесообразно использовать тогда, когда по одному запросу необходимо периодически осуществлять работу с данными при изменяющихся значениях в критерии поиска. При формировании запросов с параметрами для указания критерия отбора используются квадратные скобки.

Пусть в базе данных TYR требуется находить туры стоимостью меньше заданной цены, причем задаваемая цена меняется.

В заданный ранее запрос необходимо внести изменения: в условие отбора вместо выражения <800 следует ввести выражение в квадратных скобках <[Предельная цена тура].

В результате выполнения этого запроса на экране появится диалоговое окно *Введите значение параметра*, в которое необходимо внести предельное значение цены.

После нажатия кнопки *ОК* на экране появится таблица с информацией, удовлетворяющей заданному запросу.

Задание 25. Создать запросы, в результате выполнения которых:

выводится информация о названии авиакомпании и кодах туров, которые она обслуживает (название авиакомпании пользователь задает самостоятельно при запуске запроса);

выводится фамилия и инициалы менеджера, а также код, название и цена тура, который он обслуживает.

Задание 26. Создать запрос, отображающий информацию о турах, не требующих оформления визы. Запрос должен содержать поля: название страны, виза, название тура, цена, продолжительность тура. Сохранить запрос с именем *Для отчета*.

Создание запросов с вычислениями

Существует ряд вычислений, которые можно выполнить в запросе, например, найти сумму или среднее по значениям одного поля, перемножить значения двух полей или вычислить дату, отстоящую от текущей на три месяца. В запросах можно выполнять вычисления следующих типов:

Встроенные вычисления, называемые «итоговыми», для расчета следующих значений по группам записей или по всем записям, отобранным в запросе: сумма, среднее, число значений, минимальное или максимальное значение, стандартное отклонение или дисперсия.

Пользовательские вычисления для выполнения расчетов с числовыми и строковыми значениями или значениями дат для каждой записи с использованием данных из одного или нескольких полей. Для ввода таких выражений необходимо создать новое вычисляемое поле непосредственно в бланке запроса.

Статистические функции

Элемент	Результат	Тип поля
Sum	Сумма значений поля.	Числовой, Дата/время, Денежный, Счетчик
Avg	Среднее от значений поля.	Числовой, Дата/время, Денежный, Счетчик
Min	Наименьшее значение поля.	Текстовый, Числовой, Дата/время, Денежный, Счетчик

Max	Наибольшее значение поля.	Текстовый, Числовой, Дата/время, Денежный, Счетчик
Count	Число значений поля без учета пустых (Null) значений.	Текстовый, Числовой, Дата/время, Денежный, Счетчик, Логический, Поле объекта OLE
StDev	Среднеквадратичное отклонение от среднего значения поля.	Числовой, Дата/время, Денежный, Счетчик
Var	Дисперсия значений поля.	Числовой, Дата/время, Денежный, Счетчик

Результаты вычислений, выводящиеся в поле, не запоминаются в базовой таблице. Вычисления снова производятся всякий раз, когда выполняется запрос, поэтому результаты всегда представляют текущее содержимое базы данных. Обновить вычисленные результаты вручную невозможно.

Результаты вычислений не должны обязательно отображаться в поле. Их можно использовать в условиях отбора для определения записей, которые выбираются в запросе, или для определения записей, над которыми производятся какие-либо действия.

Допустим, что для базы данных *TYR* необходимо рассчитать выставочную скидку с цены тура в размере 3% (данные должны быть отсортированы в порядке возрастания новой цены). Для этого создается запрос, включающий код и название тура, его цену, размер скидки в этом случае рассчитывается с помощью построителя выражений.

Задание 27. Создать запрос, вычисляемый среднюю цену по всем турам.

Виды отчетов, способы создания

Отчеты предназначены для форматированного вывода данных на печать. Источниками данных для отчетов служат таблицы, запросы или инструкции SQL. Отображаемая информация автоматически изменяется при изменении данных в таблицах, на которых она базируется. Однако формат отчета сохраняется и изменяется только в том случае, когда макет отчета будет изменен.

Основные *виды* отчетов: одноколонный (простой) отчет; многоколонный отчет; табличный отчет; отчет с группировкой данных и подведением итогов; отчет по связанным таблицам; связанный отчет, т.е. отчет, содержащий другой (подчиненный отчет); отчет слиянием с документом Word (составной документ); перекрестный отчет.

Основные *разделы* отчета: заголовок отчета (начало отчета); верхний колонтитул (печатается в начале каждой страницы); область заголовка группы (отображается перед первой записью каждой группы); область данных (основная часть отчета); область примечания группы (отображается после области данных последней записи каждой группы); нижний колонтитул (печатается в конце каждой страницы); область примечаний (печатается в конце отчета).

Отчеты можно создавать несколькими способами:

с помощью автоотчета (пользователь выбирает только источник записей и макет документа); с помощью мастера отчетов (традиционная методика пошагового создания отчетов); с помощью конструктора отчетов (отчет полностью формируется пользователем).

Создание автоотчета

Средства автоматического проектирования реализованы автоотчетами.

Для создания автоотчета необходимо выбрать *Отчеты* → *Создание отчета с помощью мастера*. Так же необходимо выбрать таблицу или запрос, которые будут выступать в качестве источника данных.

При создании отчета доступны следующие методы:

Конструктор. Новый отчет создается вручную.

Мастер отчетов. Мастер MS Access сопровождает процесс создания отчета.

Автоотчет: в столбец. Создается отчет, который отображает поля из таблицы в одном столбце.

Автоотчет: ленточный. Создается отчет, который отображает данные в табличном

формате, аналогичном электронной таблице.

Диаграмма. Мастер сопровождает процесс вставки диаграммы в отчет.

Почтовые наклейки. Мастер сопровождает процесс создания отчета, форматированного для печати почтовых наклеек.

Задание 28. Создать автоотчет по таблице *Менеджер* (выбрать метод *Автоотчет: ленточный*).

Создание отчета с помощью мастера

Мастер создания отчетов работает в восемь этапов:

1 этап. Выбор базовых таблиц или запросов, на которых базируется отчет.

2 этап. Выбор полей, отображаемых в отчете.

3 этап. Выбор вида представления данных.

4 этап. Выбор полей группировки.

5 этап. Выбор порядка сортировки и вычисления, выполняемые для записи.

6 этап. Выбор вида макета для отчета.

7 этап. Выбор требуемого стиля.

8 этап. Задание имени отчета.

Задание 29. Создать отчет, основанный на запросе *Для отчета*. Поля для отчета выбрать в следующей последовательности: название, виза, название тура, цена, продолжительность тура. Вид представления данных и группировку оставить без изменения. Сортировку осуществить по оплюю *Цена* по убыванию. Вид макета выбрать *Структура 1*. Стиль выбрать *Обычный*. Сохранить отчет с именем *Страна*.

Создание отчета в режиме конструктора

Для создания отчета в режиме конструктора необходимо выбрать объект *Отчеты*. Выбрать режим *Создать отчет в режиме конструктора* и нажать кнопку *Создать*. В появившемся окне *Новый отчет*, в котором выбрать *Конструктор* (для самостоятельного создания отчета), указать источник данных – таблицу или запрос и нажать кнопку *ОК*.

Окно отчета в режиме конструктора с заголовком *Отчет1: отчет* и со следующими областями: *Верхний колонтитул; Область данных; Нижний колонтитул.*

Панель элементов, содержащая кнопки для создания элементов управления, которые можно включить в отчет. Панель элементов можно закрыть или вывести, выполнив щелчок по кнопке  на панели инструментов или выполнив команду *Вид* → *Панель элементов.*

Список полей базовой таблицы или запроса. Список полей можно вывести или закрыть, выполнив команду *Вид* → *Список полей,* или выполнив щелчок по кнопке  на панели инструментов.

Дополнительно можно вывести окно свойств создаваемого отчета, выполнив команду *Вид* → *Свойства*, или выполнив щелчок по кнопке  на панели инструментов.

Задание 30. Создать простой отчет для таблицы *Тур*, содержащий:

список сотрудников предприятия с полями: Код тура, Название тура, Продолжительность тура, Цена;

вычислить количество туров в каждой группе;

среднюю продолжительность туров

суммарную стоимость всех туров.

Для создания такого отчета выполнить:

1. Установить размеры отчета:

Переместить правую границу окна создания отчета с помощью указателя мыши так, чтобы на верхней линейке было видно число 19 (размер отчета 18 см);

Выполнить *Файл* → *Параметры страницы*. При выбранной вкладке *Страница* установить книжную ориентацию листа и размер бумаги А4. При выбранной вкладке *Поля* установить размеры левого и правого поля по 10 мм. При выбранной вкладке *Столбцы* установить: количество столбцов – 1; ширина столбца – 18 см; высота – 3 см.

2. Добавить в бланк отчета области *Заголовок отчета* и *Примечание отчета.* Для этого выполнить команду *Вид* → *Заголовок/Примечание отчета.*

3. Переместить из таблицы Тур в *Область данных* список нужных полей:

В окне таблицы *Тур* выделить в комбинации с клавишей *Ctrl* поля Код тура, Название тура, Продолжительность тура, Цена и отбуксировать их в *Область данных*. В *Области данных* появятся связанные элементы управления, т.е. элементы, связанные с полями таблицы *Тур* (слева – подпись, справа – значение поля). Выполнить щелчок мышью на свободном пространстве в области данных, чтобы убрать выделение вставленных элементов управления.

Можно просмотреть содержимое отчета на данном этапе, выбрав *Файл* → *Предварительный просмотр*. В дальнейшем можно использовать эту команду для просмотра содержимого отчета после внесения каких-либо изменений.

Переместить заголовки столбцов в область *Верхний колонтитул* для этого выделить подписи элементов управления (слева) в *Области данных*, для чего нажать клавишу *Shift* и выполнить щелчок на каждой подписи (или обвести их слева направо с нажатой левой кнопкой мыши). Выполнить команду *Вырезать*. Активизировать *Верхний колонтитул* щелчком мыши по заголовку и выполнить команду *Вставить*. Подписи будут вставлены в область *Верхнего колонтитула*.

Расставить заголовки столбцов следующим образом: подпись *Код тура* переместить в левый верхний угол области. Остальные подписи расставить так, чтобы расстояние между левыми границами подписей было равно 3см.

Выполнить редактирование и форматирование заголовков столбцов. Для этого выделить все подписи в строке (поместить курсор мыши слева от строки, чтобы курсор принял форму стрелки, направленной вправо и выполнить щелчок мышью), щелкнуть правой кнопкой мыши на выделении, в появившемся окне выбрать команду *Свойства* и установить во вкладке *Макет* следующие значения: Ширина – 3см, Высота –1см, Тип границы – Сплошная, Размер шрифта – 12см.

Уменьшить размер области *Верхний колонтитул* по размеру высоты заголовков столбцов, добавив приблизительно 0,5 см, переместив границу следующей области *Область данных* вверх;

Разместить поля в *Области данных* в ряд под заголовками соответствующих столбцов. Уменьшить размер *Области данных* мышкой переместив нижнюю границу области вверх.

4. Определить поля, по которым будет производиться группировка и сортировка данных:

Выполнить щелчок по кнопке  на панели инструментов или выбрать команду *Вид* → *Сортировка и группировка*. Открывается окно *Сортировка и группировка*.

В столбце *Поле/выражение* открыть список полей и выбрать поле *Код страны*. В столбце *Порядок сортировки* установить порядок сортировки по возрастанию.

Выполнить установку свойств в области *Свойства группы*: для *Заголовок группы* и *Примечание группы* установить значение *Да*. При этом в окне отчета появляется дополнительная область *Заголовок группы «Код страны»* перед *Областью данных* и *Примечание группы* после *Области данных*; для группировки по первому символу установить в строке *Группировка* значение *По полному значению*; в строке *Не разрывать* установить значение *Вся группа*; закрыть окно *Сортировка и группировка*.

5. Вставить в область *Заголовок группы* бланка отчета текст «*Группа туров по стране*», а затем должен выводится код страны:

Выделить область *Заголовок группы* в бланке отчета (щелкнуть на заголовке области).

Из окна *Список полей* перетащить поле *Код страны* в бланк отчета в область *Заголовок группы*.

Поместить указатель с изображением руки с вытянутым указательным пальцем на маркер, расположенный в левом верхнем углу левого поля (подпись) и отбуксировать это поле в левый верхний угол области *Заголовка группы*, отступив 0,25 см слева. Аналогично переместить правое поле вправо на 8 см от левой границы области.

В левом поле набрать текст «*Группа туров по стране»* и нажать клавишу *Enter*. Установить параметры форматирования: курсив, размер 10.

6. В окне *Панель элементов* щелкнуть по кнопке  , переместить курсор в область *Заголовок группы* под набранный текст и провести линию, подчеркнув оба поля.

7. Вставить в область *Примечание группы* бланка отчета текст «*Количество в группе*», а затем должно выводиться количество туров, относящихся к данной группе:

Создать элемент управления. Для этого выполнить щелчок на *Панели элементов* по кнопке  , а затем в области *Примечание группы* в месте расположения элемента. Появляется элемент, состоящий из 2-х частей. Переместить правую часть элемента вправо. В левую часть поля (подпись) ввести текст «*Количество в группе*». В правую часть элемента ввести формулу =*Count([Код тура])*. Произвести форматирование.

8. Вставить в область *Примечание группы* бланка отчета текст «*Средняя продолжительность тура*», а затем должна выводится средняя продолжительность всех туров, относящихся к данной группе.

Создать элемент управления. В левую часть элемента ввести текст «*Средняя продолжительность тура*». В правую часть элемента ввести формулу = *Avg([Продолжительность тура])*. Произвести форматирование.

9. Вставить в область *Примечание группы* бланка отчета текст «*Суммарная стоимость всех туров*», а затем должно выводиться суммарная стоимость всех туров, относящихся к данной группе:

Создать элемент управления. В левую часть элемента ввести текст «*Суммарная стоимость всех туров*». В правую часть элемента ввести формулу =*Sum([Цена])*. Произвести форматирование.

10. Вставить текст заголовка отчета и рядом дату создания отчета:

На панели элементов выбрать кнопку  , переместить курсор в область *Заголовок отчета*.

Отодвинуть правое поле ближе к правой границе области.

В левое поле ввести текст заголовка.

В правом поле с надписью *Свободный* ввести формулу =*Date()*.

Отформатировать поля.

11. Вставить номер страницы в области *Нижний колонтитул*:

Добавить элемент управления: в левое поле ввести текст «*Страница*», а в правое – ввести формулу =*Page*.

Задание 31. Сохранить отчет с именем Отчет с вычислениями.

Создание форм

Формы являются основным средством создания диалогового интерфейса приложения пользователя. Форма может создаваться для ввода и просмотра взаимосвязанных данных базы на экране в удобном виде.

Для создания формы необходимо выбрать *Форма* → *Создать*. На экране появится диалоговое окно *Новая форма*, в котором программа предлагает пользователю выбрать способ создания формы.

В Microsoft Access существует следующие способы создания форм.

Автоформа. Автоматическое создание формы с использованием одного из стандартных шаблонов.

Мастер форм. Создание формы с помощью мастера (в зависимости от назначения формы мастер предлагает на выбор стандартные шаблоны и стили оформления).

Конструктор. Создание формы на основе пустого бланка при помощи инструментальных средств конструктора форм.

Мастер диаграмм. Создание формы с диаграммой на основе выбранных полей таблицы.

Мастер сводных таблиц. Создание сводной таблицы Microsoft Excel на основе таблиц или запросов MicrosoftAccess.

Формы создаются на основе таблиц и запросов. При каждом открытии сохраненной формы обновляются данные запроса, на основе которого создается форма. Благодаря этому содержимое формы всегда соответствует информации в таблицах и запросах. Формы исполь-

зуют те же поля, что и таблицы, поэтому связи между форматом ввода и управления данными в этом случае не нарушаются. Также при создании формы необходимо указать таблицу или запрос, на основе которых будет создана форма.

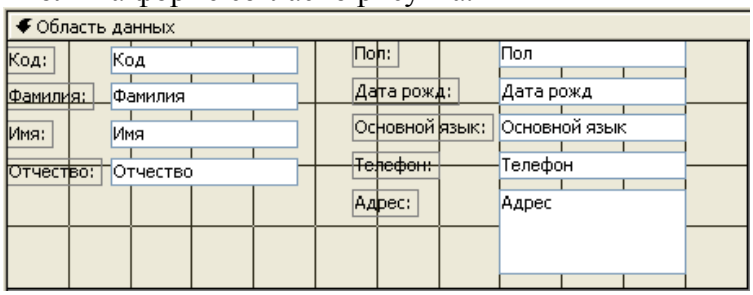
Задание 32. Создать простую форму для таблиц *Менеджер.*

Для создания простой формы необходимо выполнить последовательность следующих действий:

1. Выбрать *Формы* → *Создать*. В диалоговом окне *Новая форма* выбрать вариант создания с помощью конструктора, а в качестве источника данных – таблицу *Менеджер.* Появится окно *Форма 1: форма* в режиме конструктора совместно с окном *Список полей*. Если на экране отсутствует список полей выбранной для построения формы таблицы, то необходимо выбрать пункт меню *Вид* → *Список полей*.

2. Переместить поля из списка на форму (по одному или предварительно выделив с использованием клавиши *Shift* и мыши; для выделения всех полей выполнить двойной щелчок мышью на заголовке окна *Список полей*).

3. Разместить поля на форме согласно рисунка.



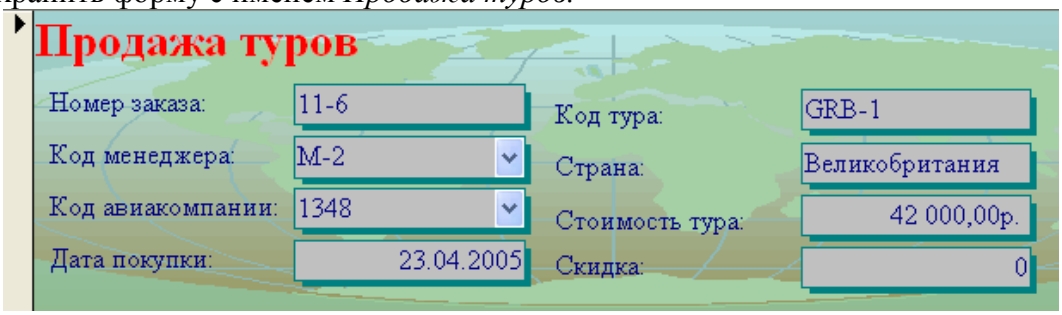
4. Применить для формы *Автоформат* → *Промышленный.*

5. С помощью инструмента Aa вставить в центре формы заголовок *Менеджеры туристической фирмы* (размер шрифта – 14, начертание – жирный, цвет текста – черный, цвет заливки – прозрачный).

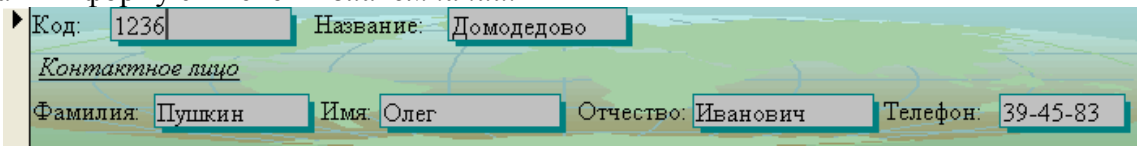
6. Перейти из режима конструктора в режим просмотра, щелкнув по кнопке – *Просмотр формы*. Если созданная форма не соответствует рисунку, то вернуться в режим конструктора и внести изменения.

7. Сохранить форму с именем *Менеджеры*.

Задание 33. Создать простую форму для таблицы *Продажа туров.* Поля на форме разместить в соответствии с рисунком. Выбрать для формы *Автоформат* → *Международный.* Сохранить форму с именем *Продажа туров.*

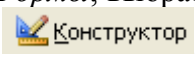


Задание 34. Создать простую форму для таблицы *Авиакомпании.* Поля на форме разместить в соответствие с рисунком. Выбрать для формы *Автоформат* → *Международный.* Сохранить форму с именем *Авиакомпании.*



Задание 35. Создать объединенную форму, включающую две ранее созданные: *Продажа туров* и *Авиакомпании*.

Для создания объединенной формы необходимо выполнить последовательность следующих действий:

1. В окне базы данных, при выбранной вкладке *Формы*, выбрать форму *Продажа туров*. Она будет основной. Выполнить щелчок по кнопке 

.

2. В нижней части формы *Продажа туров* разместить заголовок *Авиакомпании* (размер шрифта – 24, цвет шрифта – красный).

3. Расположить окна базы данных и Конструктора с открытой формой *Продажа туров* таким образом, чтобы они не перекрывали друг друга.

4. В окне базы данных выбрать форму *Авиакомпании*. Переместить ее в окно формы *Продажа туров* на свободное место, под заголовком *Авиакомпании*.

5. Сохранить форму с именем *Продажа туров для заполнения таблицы*.

Задание 36. Перейти к вкладке *Таблицы*, выбрать таблицу *Авиакомпании* и удалить все записи из таблицы, кроме столбца *Код*. Заполнить таблицу *Авиакомпании*, используя для заполнения созданную форму *Продажа туров для заполнения таблицы*. Просмотреть заполненную таблицу *Авиакомпании* в режиме *Таблица.* Убедиться, что все записи, помещенные в таблицу, верны. При необходимости внести изменения в данные таблицы. Закрыть таблицу. Подтвердить сохранение произведенных изменений.

Литература:

1. Информатика: учеб.: рек. Мин. обр. РФ / под ред. Н. В. Макаровой. – М.: Финансы и статистика, 2000, 2005, 2001. – 268 с.

2. Информатика. Базовый курс: учеб.: рек. Мин. обр. РФ / Ред. С.В. Симонович. – СПб.: Питер, 2000, 2004, 2005, 2006. – 638 с.

3. Методические указания к практическим занятиям

В процессе обучения студент должен прослушать определенный теоретический материал и закрепить этот материал на практических занятиях, а также при выполнении домашних самостоятельных работ.

Практическое занятие должно начинаться с проверки домашнего задания. При этом допустимо некоторые, наиболее сложные задачи, с которыми не справилась большая часть студентов решить на доске. Тем самым создается прочная база для дальнейшего обучения.

При изучении новой темы необходимо постоянно обращаться к теоретическому материалу. Иногда теория оказывается заданной на самостоятельное изучение. В этом случае преподаватель-практик обязан помочь студенту в выборе литературы, разъяснить трудные и непонятные места в тексте, ответить на все вопросы. Переходить к практическим задачам возможно только после полного усвоения теории. Недопустимо повторять чтение лекции на практике, если студенты забыли конспекты лекций и не помнят их суть.

При решении задач нужно обосновать каждый этап решения исходя из теоретических положений курса. Если студент видит несколько путей решения, то необходимо помочь ему выбрать наиболее рациональный. Решение каждой задачи должно доводиться до ответа, требуемого условием.

Для оптимизации учебного процесса и развития практических навыков овладения математикой весьма эффективным является проведение кратких самостоятельных работ, как по практическому, так и по теоретическому материалу. При этом целесообразно формулировать вопросы по теории таким образом, чтобы для ответа не требовались долгие и сложные дока-

зательства и выводы. Такая форма контроля позволяет выявить наличие и прочность базовых знаний по изучаемой теме. Аналогично, практические задания должны быть составлены предельно просто и ясно. При проведении таких кратких работ студенты не должны пользоваться никаким справочным материалом.

В конце занятия необходимо подвести итог, объявить тему и план следующего занятия, задать домашнее задание, указав литературу, которой желательно воспользоваться при его выполнении.

4. Методические указания по выполнению курсовых работ

Курсовая работа является важнейшим элементом самостоятельной работы студентов.

Цель курсовой работы: углубленная разработка одной из проблем курса, представляющейся актуальной и недостаточно исследованной, либо требующей переосмысления в новых условиях.

Задачи подготовки курсовой работы:

систематизация, закрепление и расширение теоретических и практических знаний;

приобретение опыта применения этих знаний при решении конкретных задач;

овладение элементами методики научного исследования;

формирование интереса к научно-исследовательскому поиску;

развитие и совершенствование самообразовательных потребностей и умений;

выработка самостоятельной позиции, умений ее отстаивать и защищать.

Курсовая работа должна показать умение слушателя самостоятельно изложить проблему, выявить наиболее приоритетные вопросы, применить элементы исследования, или представить собственные экспериментальные или опытные данные.

Курсовая работа не может быть простой компиляцией и состоять из фрагментов различных статей и книг. Она должна быть научным, завершенным материалом, иметь факты и данные, раскрывающие взаимосвязь между явлениями, процессами, аргументами, действиями и содержать нечто новое: обобщение обширной литературы, материалов эмпирических исследований, в которых появляется авторское видение проблемы и ее решение. Этому общетеоретическому положению подчиняется структура курсовой работы, ее цель, задачи, методика исследования и выводы.

Курсовая работа является квалификационным учебно-научным трудом студента, посвященным самостоятельной разработке избранной проблемы.

Курсовая работа должна состоять из титульного листа, оглавления, введения, основной части, заключения, списка использованной литературы и приложений (если они есть).

Титульный лист как первая страница работы должен содержать следующие реквизиты: названия учебного заведения, кафедры, темы работы, фамилию, имя, отчество автора, курс и номер его группы, фамилию, инициалы, ученую степень и звание научного руководителя, место и год выполнения работы.

Следующей страницей оформляется оглавление. Оно должно включать все заголовки в работе и номера страниц, с которых они начинаются.

Введение объемом 1,5-2 страницы призвано познакомить читателя с сущностью исследуемой темы. Во введении указываются актуальность темы, степень ее разработанности в литературе, формулируются цели работы и ее предмет, характеризуются использованные автором материалы. Во введении целесообразно объяснить, почему именно под таким углом зрения раскрывается тема, почему отдельным вопросам уделяется особое внимание, а другие излагаются более поверхностно.

Основная часть курсовой работы излагается последовательно в соответствии с оглавлением (планом). Все параграфы работы должны быть логически связаны между собой и в совокупности раскрывать тему. После каждого параграфа желательно формулировать краткие выводы.

В основной части работы необходимо отразить использование источников. При этом не допускается переписывание текста из учебников или другой литературы. Должна быть произведена творческая обработка материала. Важнейшие теоретические положения темы

излагаются своими словами и при необходимости подкрепляются цитатами. Цитаты оформляются в соответствии с библиографическими правилами и сопровождаются постраничными ссылками на используемый источник с указанием страниц. Примеры таких ссылок можно увидеть практически в каждом печатном издании по уголовному процессу.

При работе с литературой рекомендуется находить проблемные ситуации, противоречивые взгляды. Различные позиции авторов желательно отразить в содержании работы, изложить аргументы в их критику и поддержку. После анализа точек зрения о дискуссионных вопросах рекомендуется изложить и собственную позицию.

За основной частью работы следует заключение. В заключении подводятся итоги работы в целом, формулируются выводы, отражающие степень достижения поставленных целей. Содержание заключения последовательно и логически стройно представляет результаты всей курсовой работы. Примерный объем заключения не превышает 1,5-2 страницы.

Список использованной литературы является важнейшей частью курсовой работы, поскольку отражает проделанную работу и глубину исследования темы. В список должны быть включены только те источники, которые действительно использовались автором и на которые есть ссылки в тексте работы. В начале списка необходимо указать нормативные акты по их юридической силе. После этого в алфавитном порядке перечисляются монографии, пособия, статьи, комментарии и т. д.

Курсовая работа при условии наличия всех ее структурных частей (введения, глав, заключения и списка литературы) в отведенные для этого сроки может быть отдана научному руководителю на проверку с возможностью последующей доработки (устранения замечаний). Научный руководитель возвращает курсовую работа на доработку с указанием конкретных замечаний или направлений существенного улучшения курсовой работы, либо принимает курсовую работу к защите.

В случае представления курсовой работы преимущественно реферативного характера, в которой не удается выделить личный вклад автора, либо сдачи курсовой работы, в которой не выполнены основные требования к содержанию, структуре и оформлению курсовая работа не рассматривается, а возвращается с указанием данных причин.

Курсовая работа сдается научному руководителю с возможностью ее доработки только один раз.

В случае возврата научным руководителем курсовой работы с замечаниями, ее доработка осуществляется в отведенные сроки. Если часть сделанных замечаний непонятна студенту ему настоятельно рекомендуется лично прийти на консультацию к научному руководителю для их разъяснения. После устранения сделанных замечаний курсовая работа сдается руководителю для окончательной проверки и оценивания. Обязательно при этом прикладывается список сделанных на предыдущем этапе научным руководителем замечаний. Студент может отказаться от устранения сделанных замечаний, если его курсовая работа может претендовать на положительную оценку.

На защиту курсовой работы студенты дневного отделения приходят с зачетной книжкой, а студенты иной формы обучения – с зачетной книжкой и паспортом.

Защита представляет собой устное собеседование, по результатам которого определяется уровень знаний автора по теме курсовой работы, а также по объекту и предмету исследования, умение автора ясно и обоснованно излагать свои мысли и отвечать на вопросы. Слово «защита» подразумевает, что автор в ходе устного собеседования защищает (обосновывает) свою точку зрения на рассматриваемые вопросы и полученные им в курсовой работе выводы и результаты.

Итоговая оценка за курсовую работу определяется с учетом уровня защиты и оценки за текст курсовой работы, поставленной научным руководителем, и проставляется в зачетную книжку и в ведомость. Неудовлетворительная оценка за текст курсовой работы означает недопуск к защите и необходимость доработки курсовой работы до приемлемого состояния, неудовлетворительная оценка за защиту курсовой работы означает необходимость писать новую курсовую работу по новой теме. Итоговая оценка определяется на основе балльно-

рейтинговой системы следующим образом.

Порядок расчета баллов за курсовую работу в балльно-рейтинговой системе

№	компетенция	максимальное количество баллов
Текст курсовой работы		
1	своевременность выполнения основных этапов написания курсовой работы, соответствие оформления основным требованиям	10
2	умение грамотно формулировать проблему курсовой работы, свои мысли и выводы, соответствие структуры работы ее задачам	10
3	качество изучения теоретических аспектов рассматриваемой проблемы	10
4	качество проведенного анализа отраслевого рынка	20
5	соответствие проведенного анализа и полученных результатов поставленным целям и задачам	20
6	наличие и глубина авторского вклада в решение поставленных целей и задач	20
	Итого – баллов за текст курсовой работы	80
Защита курсовой работы		
1	качество ответов на поставленные вопросы	10
2	умение формулировать основные выводы и результаты, полученные в курсовой работе в соответствии с поставленными целью и задачами, обосновывать и защищать их	10
	Итого – баллов за защиту курсовой работы	20
Всего баллов		100

Итоговая оценка определяется по следующей шкале.

Порядок определения итоговой оценки за курсовую работу на основе набранных баллов

Диапазон набранных баллов						Оценка по пятибалльной шкале
Всего		в том числе:				
		текст		защита		
98	100	78	80	20	20	отлично
94	97	75	78	19	19	
90	93	72	74	18	19	
86	89	69	71	17	18	
83	85	66	68	17	17	хорошо
80	82	64	66	16	16	
77	79	62	63	15	16	
74	76	59	61	15	15	
70	73	56	58	14	15	удовлетворительно
66	69	53	55	13	14	
63	65	50	52	13	13	
60	62	48	50	12	12	
50	59	40	47	10	12	
25	49	20	39	5	10	неудовлетворительно
0	24	0	19	0	5	неудовлетворительно без возможности пересдачи

Тематика курсовых работ

1.Программа подсчета стоимости поездки. Город разбит на 5 секторов; поездка из одного сектора в другой (и внутри секторов) имеет фиксированную цену. Пользователь вводит сектор отправления и сектор назначения. Программа определяет стоимость поездки с учетом

количества пассажиров и типа автомобиля. Предусмотреть возможность просмотра таблицы цен.

2.Программа подсчета стоимости покупок семьи. Пользователь вводит список покупок с указанием цены и покупателя. Программа подсчитывает общую сумму всех покупок и количество потраченных денег каждым членом семьи. Предусмотреть возможность просмотра списка покупок для каждого члена семьи

3.Программа «Гороскоп». Пользователь водит дату своего рождения, ему выдаются его гороскопы по знаку зодиака и по восточному календарю, подсчитывает, сколько дней прожил пользователь. Предусмотреть возможность просмотра списка всех пользователей этой программы (в текущем сеансе), возможность выбора всех пользователей конкретного знака гороскопа.

4.Программа «виртуальный магазин». Пользователь вводит сумму, которую собирается потратить, ему предлагаются товары, которые он может приобрести на эту сумму (список товаров задается в программе). Если после выбора покупателем какого-либо товара, у него остаются деньги, программа снова предлагает ему список товаров, которые он может приобрести. Цикл повторяется до тех пор, пока покупатель не потратит все деньги или не захочет закончить покупки. Предусмотреть возможность просмотра списка покупок

5.Программа «Калькулятор». Пользователь вводит арифметическое выражение, программа рассчитывает результат. Предусмотреть возможность ввода как с клавиатуры, так и с помощью «мыши» путем нажатия на соответствующие кнопки на форме.

6.Игра «Крестики-нолики». Два игрока по очереди делают ходы на поле 5*5. программа оценивает каждый ход и выдает информацию: продолжение игры или выигрыш игрока.

7.Программа для продажи билетов в кинотеатре. Небольшой кинотеатр на 10 рядов, 20 мест в каждом ряду. Программа показывает таблицу посадочных мест, пользователь отмечает крестиком проданные места. Программа определяет, сколько человек еще можно поместить, можно ли поместить рядом группу из нескольких человек, подсчитывает количество проданных билетов и общую сумму выручки с учетом разницы в стоимости билетов (VIP-места, боковые места, центр). Стоимость билетов задается в программе.

8.Программа «Автостоянка». Стоянка на 20 машин. Пользователь заносит в программу машины, приехавшие на стоянку, учитывая, что большие машины занимают 2 места. Программа определяет, можно ли поместить на стоянку машину, сколько осталось мест и общую выручку с учетом разной стоимости места для разных машин.

9.Программа «Проверка знаний» по арифметике. Пользователю предлагается пройти тест из 20 примеров на 4 арифметических действия (сложение, вычитание, умножение, деление) над однозначными числами. Числа и вид действия определяется случайным образом. Программа выставляет оценку и определяет, какое из арифметических действий пользователь знает хуже.

10. Программа «Прогноз погоды». Пользователю задаются несколько вопросов (например, по народным приметам), подразумевающих ответы «Да» или «Нет». По ответам программа определяет, какая погода будет завтра. Каждый последующий вопрос зависит от ответа на предыдущий.

11. Программа «Расписание полетов». В программе хранится расписание рейсов самолетов с указанием цен на билеты. Пользователь вводит место отправления, место назначения, дату отлета, дату прилета (например, не позднее такой-то даты). Программа выбирает все возможные маршруты (прямые рейсы, рейсы с 1 пересадкой), подсчитывает их стоимость.

12. Программа «Продажа билетов на поезд». Поезд идет от места отправления к месту назначения с 1 остановкой. На остановке люди могут зайти и выйти. Программа распределяет пассажиров по купе с учетом пожеланий (семьи хотят ехать вместе и т.д.). Если программа не может разместить пассажиров как они желают, но при этом мест в вагоне хватает, предлагает возможные варианты.

13. Тест Люшера. Пользователю показываются 8 разноцветных карточек, расположенных в случайном порядке. Пользователь выбирает карточки в порядке уменьшения приори-

тетов. Выбранная карточка исчезает. В зависимости от порядка выбора карточек выдается результат теста.

14. Программа «Кулинарный рецепт». В программе заданы цены на основные продукты. Пользователь указывает, сколько блюд будет готовить (1-3) , выбирает из списка продукты, необходимые ему по рецепту для каждого блюда, указывает количество продукта. Программа выдает меню, стоимость каждого блюда и стоимость всех блюд.

15. Программа «Биоритмы». Существует 3 цикла: физический (период – 23 дня), эмоциональный (период – 28 дней) и интеллектуальный (период – 33 дня). Начало циклов совпадает с датой рождения. Половину периода человек испытывает соответствующий подъем, вторую половину периода – спад. Пользователь вводит дату рождения, программа подсчитывает количество прожитых им дней и определяет физическое, эмоциональное и интеллектуальное состояние пользователя.

16. Конвертер единиц измерения. Программа позволяет переводить одни единицы измерения в другие, выполнять вычисления с данными в разных единицах измерения (например, к расстоянию в метрах прибавить расстояние в ярдах и получить результат в милях).

17. Программа «Экранная клавиатура». Программа имитирует клавиатуру на экране и позволяет писать тексты при помощи «мыши».

18. Программа «Тест памяти». Программа предлагает пользователю несколько картинок в случайном порядке, а затем просит воспроизвести порядок их появления и оценивает память тестируемого.

19. Программа «Текстовый редактор». Программа имитирует текстовый редактор, т.е. позволяет в специальном поле набирать текст, сохранять его в формате txt, загружать из текстового файла, выполнять стандартные операции: поиск и замена, копирование в буфер и вставка из буфера, применение шрифтов (ко всему тексту).

5. Методические указания по самостоятельной работе студентов

Для закрепления полученных теоретических и практических знаний студентам в течение всего учебного года предлагаются индивидуальные задания для самостоятельной работы. Консультирование по выполнению индивидуальных заданий проводится как непосредственно в компьютерных классах (во время консультаций), так и через электронный обмен сообщениями, посредством Интернет. Защита индивидуальных заданий по темам может проводится в виде Круглого стола, когда каждый студент выступает с презентацией выполненной работы, а преподаватель вместе с остальными студентами оценивает работу. Задания по темам также могут быть выданы студентам в качестве домашних заданий в виде электронных файлов. Контроль выполненных заданий осуществляется либо непосредственно на занятиях, либо на консультациях.

В рабочей программе п.6 представлены виды самостоятельной работы по каждой теме дисциплины и трудоемкость в часах.

Индивидуальная самостоятельная работа включает две части: текстовую (реферат), подготовленную с использованием текстового процессора MS Word, и электронную презентацию, выполненную средствами MS Power Point. Текстовая часть должна быть представлена в виде твердой копии на бумажном носителе и в электронном виде (на дискете или CD, в формате RTF).

Защита индивидуальных заданий будет происходить на практическом занятии в аудитории. Каждое выступление должно сопровождаться презентацией.

Тематика домашних заданий

1. Norton Commander. Описание и возможности.
2. Автоматизированные системы обработки информации.
3. Архитектура персональных компьютеров.
4. Базы данных в Internet.
5. Внешние устройства ПК. Функциональные возможности. Основные характеристики.
6. Данные и информация.

7. Дисковая система IBM PC.
8. Информационные технологии и их роль в обществе.
9. Информационное обеспечение как необходимая услуга для функционирования экономики в современных условиях.
10. Информационные системы в экономике.
11. Как появились компьютеры.
12. Компьютерные вирусы.
13. Локальные и глобальные сети. Электронная почта.
14. Многопользовательская система Windows NT.
15. Модель файловой системы FAT.
16. Накопители и носители информации, жесткие диски.
17. Основные понятия мультимедиа.
18. Персональные компьютеры в медицинской практики.
19. Приводы CD-ROM. Форматы и стандарты.
20. Применение компьютера в туристической деятельности.
21. Принцип работы CD-ROM.
22. Развитие архитектуры материнских плат для PC.
23. Система автоматизированной обработки статистической информации.
24. Социальная информатика.
25. Сравнение операционных систем DOS, UNIX, OS/2, WINDOWS.
26. Теория и практика производства накопителей на гибких магнитных дисках.
27. Файловая система.
28. Экспертные системы. Классификация экспертных систем. Разработка простейшей экспертной системы.
29. Электронная почта.
30. Проблема моделирования на ЭВМ основных функций человеческого мышления.
31. Информационный маркетинг в Интернете.
32. Общая характеристика преступлений в сфере компьютерной информации.
33. Компьютерные преступления.
34. Компьютерная преступность и компьютерная безопасность.
35. Современные банковские автоматизированные системы.
36. Компьютерные технологии в строительстве.
37. Информационная безопасность.
38. Технологии создания сетей ЭВМ.
39. DVD-ROM устройство и принцип работы.
40. Электронная коммерция.

Требования к оформлению

1. Поля страницы: слева – 3, сверху и снизу – 2, справа – 1.
2. Размер шрифта – 14.
3. Межстрочный интервал – 1,5.
4. Шрифт – Times New Roman.
5. Нумерация страниц – внизу, по центру.
6. В конце – список использованной литературы.
7. Объем реферата – не менее 15 листов.

IV. КОНТРОЛЬ ЗНАНИЙ

В течении семестра знания студентов оцениваются с использованием рейтинговой системы, которая складывается из оценки за работу в семестре – максимально 60 баллов и экзаменационной оценки – максимально 40 баллов. Максимально возможное количество равно 100. Баллы по разным модулям приведены в рабочей программе п.12.

Минимальное количество баллов в семестре, необходимое для получения студентом допуска на экзамен, равно 30 баллов, на зачет – 40 баллов.

Минимальное количество баллов за выполнение экзаменационной работы, необходимое для получения оценки: «удовлетворительно» - 15 баллов, «хорошо» - 20 баллов, «отлично» - 30 баллов.

Соответствие итогового рейтинга студента и традиционных оценок устанавливается по следующей шкале: «неудовлетворительно» - 0-50 баллов, «удовлетворительно» - 51-75 баллов, «хорошо» - 76-90 баллов, «отлично» - 91-100 баллов.

В качестве основных средств текущего контроля используется тестирование. В качестве дополнительной формы текущего контроля предлагаются аудиторные и внеаудиторные письменные задания (контрольные работы).

1. Текущий контроль знаний

1. Алгоритмизация и программирование. Моделирование решения функциональных и вычислительных задач.

1. Программы-компиляторы служат для:

a) автоматического перевода программы в машинный код и последующего её использования без исходного текста;

b) анализа структуры очередного оператора языка из текста программы и его исполнения перед переходом к следующему оператору языка;

c) написания текста программы с возможностью определения синтаксических ошибок;

2. Выберите верное высказывание:

a) алгоритм имеет свойство дискретность, означающее разбиение алгоритма на конкретные действия;

b) алгоритм имеет свойство специфичность, обозначающее, что алгоритм составляется для одной задачи;

c) алгоритм имеет свойство результативность, означающее, что после выполнения алгоритма должен быть получен графический результат;

3. К языкам программирования высокого уровня относятся:

a) Pascal, Basic, Си++, Ассемблер;
b) Pascal, Basic, Ассемблер, Access;
c) Basic, Си++, Pascal, Java;

4. Алгоритм – это:

a) точное предписание, определяющее процесс перехода от исходных данных к результату;

b) требования, предъявляемые к программе;

c) проведение расчетов и анализ результатов;

5. Условие – это:

a) выражение логического типа;
c) все ответы верны;

b) повторяющиеся команды;

6. Язык программирования низкого уровня – это:

a) не учитывают особенности конкретных компьютерных архитектур и позволяют разрабатывать программы с помощью понятных для пользователя команд;

b) языки, операторы которых близки к машинному коду и ориентированы на конкретные команды процессора;

c) представляют собой комплект программ, обеспечивающий возможности работы на компьютере;

7. Выберите верное высказывание:

a) каждый алгоритм решения задачи имеет входные и выходные данные;

b) каждый алгоритм содержит в себе разветвляющиеся и циклические структуры;

c) не всякий алгоритм можно представить в виде блок-схемы;

8. Выберите верное высказывание:

a) алгоритм имеет свойство дискретность, означающее разбиение алгоритма на конкретные действия;

b) алгоритм имеет свойство специфичность, обозначающее, что алгоритм составляется для одной задачи;

c) алгоритм имеет свойство результативность, означающее, что после выполнения алгоритма должен быть получен графический результат.

2. Основы программирования в Delphi.

1. Delphi реализует интерфейс управления окнами:
a) SDI (Single Document Interfase); c) RDI (Reformative DI);
b) MDI (Multiple Document Interface);
2. Окно формы используется для:
a) проектирования окна программы; c) создания текста программы;
b) выбора визуальных компонентов;
3. В инспекторе объектов задаются:
a) заголовок формы; c) программа и алгоритм;
b) свойства и события;
4. Страница Properties используется:
a) для задания свойств объекта; c) в меню главного окна;
b) для задания событий объекта";
5. TButton является:
a) командной кнопкой; c) свойством;
b) меткой;
6. Свойство Font определяет:
a) параметры шрифта; c) характеристики формы;
b) палитру фона;
7. Свойства размера объекта:
a) Width, Height; b) Width, Top; c) Left, Top;
8. TButton не имеет свойства:
a) Caption; b) Color; c) Left;
9. Событие щелчок мыши:
a) OnClick; b) OnCreate; c) OnClose;
10. Верное описание события:
a) Procedure Form1. Button1Click(); c) Procedure Button1.Click();
b) Procedure Form1.OnClick(Button1);
11. Команда прекращения программы:
a) Project/Compile; b) Run/Run; c) Run/Reset;
12. Выполняемый файл *.exe создается:
a) при компиляции сохраненной программы; b) при алгоритмизации программы;
c) во время сохранения программы;

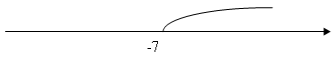
3. Линейная структура.

1. При моделировании решения задач этап алгоритмизация следует:
a) после этапа программирования;
b) после этапа тестирования;
c) после этапа выбор методологии разработки программы;
2. Компонент TEdit не имеет свойство:
a) Text; b) Color; c) Caption;
3. Переменная типа Boolean может принимать значение:
a) 10; b) False; c) 'верно';
4. Выберите верную команду:
a) edit1.text:=FloatToStr(S); c) edit1.text:=S(StrToFloat);
b) S:=FloatToInt(edit1.text);
5. Функция возведение переменной x в квадрат:
a) SQRT(x); b) SQR(x); c) не существует;
6. Функция (r mod 2) находит:

a) модуль числа; c) остаток от деления;
b) целую часть от деления;
7. Выражение ABS(6+5*x) при x=2 возвращает значение:
a) 4; b) 16; c) 22;
8. При обнаружении синтаксической ошибки курсор помещается на символ:
a) в котором обнаружена ошибка; c) перед ошибкой;
b) следующий после ошибки;
9. Наличие структурных ошибок проверяется при:
a) компиляции; c) работе с выполняемой программой;
b) закрытии окна кода программы;
10. Модуль является:
a) не обязательной программой, однако при его наличие увеличивается быстродействие;
b) основной программой, к которой подключаются файл проекта;
c) вспомогательной программой, которая присоединяется к файлу проекта;

4. Оператор ветвления.

1. В операторе ветвления выполняется:
a) сначала одна, потом другая цепочка действий;
b) только одна цепочка действий;
c) две цепочки действий при истинном условии, одна - при ложном;
2. Веточка THEN в ветвление определяет:
a) действия, выполняемые при ложности условия;
b) действия, выполняемые при истинности условия;
c) проверку условия для выполнения действия;
3. Верно записанное условие:
a) y:=5; b) y=5; c) y and 5;
4. Условие not(x>=5) при x=-20 равно:
a) boolean; b) false; c) true;
5. Выберите верное утверждение
a) не верна; c) неполного ветвления;
b) полного ветвления;
4. Условие принадлежности выделенной области в Pascal:



a) not(x<-7); c) not(x<=-7);
b) not(x>-7);
8. Для формы F1 событие щелчок мыши на компоненте CheckBox1 описывается:
a) Procedure TF1. CheckBox1Click ();
b) Procedure F1. CheckBox1 OnClick ();
c) Procedure CheckBox1 Click ();
9. Независимый переключатель позволяет:
a) включать флажки только поочередно;
b) включать одновременно любое количество флажков;
c) включать один флажок обязательно при выключенных других;
10. If (x<4)and(x>-2) then y:=x+2 else y=x-5; при x=5 вернет значение:
a) y=x-5; b) 7; c) 0.

5. Оператор выбора.

1. Оператор выбора позволяет:
a) правильно выделить соответствующую строку текста;
b) выбрать одно из нескольких продолжений программы;
c) выполнить две цепочки действий;
2. В операторе выбора промежуток значений от 1 до 10 указывается:
a) 1...10; b) 1-10; c) 1..10;
3. К классу TStrings относится свойство компонента TRadioGroup:
a) String; b) Items; c) Lines;
4. Свойство, возвращающее номер выделенной строки, у компонента Tmemo:
a) называется Items; b) отсутствует;

c) используется ItemIndex;

5. Команда присвоения n:Integer количество строк в memo1 записывается:

a) n:=memo1.Lines.Count; c) memo1.Lines.Count(n);

b) n:=memo1.Count;

6. В компоненте TListBox редактирование текста

a) допустимо после выделения радио-кнопки; b) невозможно; c) возможно;

7. Команда, скрывающее окно ListBox1, записывается:

a) ListBox1.ItemIndex.Clear;

b) ListBox1.Visible:=false;

c) ListBox1.Items.visible:=false;

8. В компоненте TRadioGroup редактирование текста

a) допустимо после выделения радио-кнопки; b) невозможно; c) возможно;

9. Команда, скрывающее окно RadioGroup1, записывается:

a) RadioGroup1.ItemIndex.Clear;

b) RadioGroup1.Visible:=false;

c) RadioGroup1.Items.visible:=false;

10. Зависимый переключатель позволяет:

a) включать переключатели только поочередно(первую, вторую и т.д.);

b) включать одновременно любое количество переключателей;

c) включать один переключатель обязательно при выключении других.

6. Циклические структуры.

1. Циклические операторы бывают:

a) с предусловием/с параметром/с постусловием;

b) простые/сложные/многовариантные;

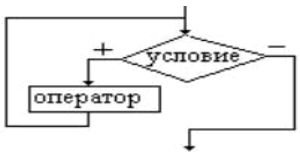
c) с начальным/повторяющимся/ конечным условиями;

2. Условие проверяется в конце в операторе:

a) Repeat; b) For; c) While;

3. Выберите верное утверждение

a) блок-схема цикла с предусловием;



b) в блок-схеме веточки true и false следует поменять местами;

c) в блок-схеме оператор2 веточки false лишний;

4. Тело цикла в цикле с предусловием выполняется:

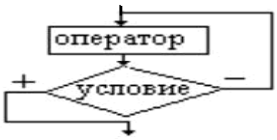
a) при ложности условия;

b) при истинности условия;

c) вне зависимости от проверки условия n раз;

5. Выберите верное утверждение:

a) блок-схема цикла с постусловием



b) в блок-схеме веточки true и false следует поменять местами;

c) в блок-схеме отсутствует второй оператор;

6. Оператор цикла с постусловием имеет вид:

a) Repeat (<условие>) do <оператор>;

b) While (<оператор>) Until <условие>;

c) Repeat (<оператор>) Until <условие>;

7. Цикл с параметром верно записан:

a) For i:=10 downto 2 do <оператор>;

b) For i:=10 to 2 do <оператор>;

c) For i:=10 do 2 downto <оператор>;

8. При решение задач циклы с параметром и постусловием различаются:

a) изменение параметра цикла происходит автоматически/оператором;

b) тело цикла выполняется n/(n-1) раз;

c) условие цикла записывается в одну/две строки;

9. В программах переменная, хранящая значение a в степени x первоначально равна:

a) -1; b) 1; c) 0;

10. a:=3; t:=1; For i:=1 to 5 do t:=t*a;

a) блок программы вычисления 5!;

b) блок программы вычисления a в 5 степени;

c) блок программы вычисления суммы ряда a+2a+3a+4a+5a.

7. База данных

1. Ключ в базе данных – это:

a) простейший объект БД для хранения значений одного параметра реального объекта или процесса;

b) поле, по которому выполняется фильтрация данных в таблице параметрам;

c) поле или совокупность полей, однозначно определяющих записи таблицы.

2. Представлена база данных «Школа»:

Запрос для вывода списка: учеников 10 классов, 1988 года рождения, имеющих оценки не ниже 4 содержит выражение:

	Фамилия	Год рождения	Класс	Оценка
	Лыкова Ольга	1988	10	5
	Семенов Олег	1987	11	4
	Морозов Иван	1987	11	3
	Рыков Роман	1988	10	5
	Попов Сергей	1988	10	4
	Зайцева Марина	1987	10	5

a) Оценка>=4 и Год рождения = 1988 и Класс = 10;

b) Класс = 10 и Год рождения = 1988 и Оценка = 5 и Оценка = 4;

c) Оценка >=4 или Год рождения = 1988 и Класс = 10.

3. Установленные связи между таблицами реляционной базы данных помогают:

a) избежать дублирования информации;

b) определить местонахождение нужной таблицы;

c) производить сортировку таблицы.

5. Система управления базами данных – это:

a) формальный аппарат ограничений на формирование таблиц, который позволяет устранить дублирование;

b) комплекс программных и языковых средств, предназначенных для создания, ведения и совместного применения баз данных многими пользователями;

c) система, реализующая сбор, обработку и манипулирование данными и включающая технические средства, программное обеспечение и соответствующий персонал.

5. Какую строку будет занимать запись Pentium II после проведения сортировки по возрастанию в поле Винчестер?

a) 1;

b) 2;

c) 4;

	Компьютер	Опер. память	Винчестер
1	Pentium	16	2Гб
2	386DX	4	300Мб
3	486DX	8	800Мб
4	Pentium II	32	4Гб

6. Тип поля (числовой, текстовой и др.) в базе данных определяется:

a) названием поля;

b) количеством строк;

c) типом данных.

7. Запись в БД Access:

a) столбцы реляционной таблицы;

b) строки реляционной таблицы;

c) одно или несколько полей, имеющих уникальное значение для реляционной таблицы.

8. Поле какого типа не может быть ключевым:

a) числовое; b) логическое; c) дата.

9. В какой модели данных предполагается, что любой элемент может быть связан с любым количеством других элементов:

a) реляционная; b) сетевая; c) иерархическая;

10. База данных, в которой данные сгруппированы в двумерные таблицы, связанные между собой, называется:

a) иерархическая; b) сетевая; c) реляционная.

11. Поисковая фраза в текстовых БД называется:

a) вопросом; b) ответом; c) запросом.

12. Представлена таблица базы данных «Кадры»:

При поиске по условию (Год рождения>1956 и оклад<5000) будут найдены фамилии: называется:

	Фамилия	Год рождения	Оклад
	Иванов	1956	2400
	Петров	1957	5300
	Сидоров	1956	3600
	Скворцов	1952	1200
	Трофимов	1958	4500

a) Иванов, Петров, Трофимов;

b) Трофимов;

c) Иванов, Петров, Трофимов, Скворцов.

2. Итоговый контроль знаний.

В рабочей программе п.9 представлены вопросы к зачету.

V. ИНТЕРАКТИВНЫЕ ТЕХНОЛОГИИ И ИННОВАЦИОННЫЕ МЕТОДЫ, ИСПОЛЬЗУЕМЫЕ В ОБРАЗОВАТЕЛЬНОМ ПРОЦЕССЕ.

Образовательный процесс по дисциплине строится на основе комбинации следующих образовательных технологий.

Интегральную модель образовательного процесса по дисциплине формируют технологии методологического уровня: модульно-рейтинговое обучение, технология поэтапного формирования умственных действий, технология развивающего обучения, элементы технологии развития критического мышления.

Рекомендуется использование информационных технологий при организации коммуникации со студентами для представления информации, выдачи рекомендаций и консультирования по оперативным вопросам (электронная почта), использование мультмедиа-средств при проведении лекционных и лабораторных занятий.

Игровые имитационные методы:

Мозговой штурм – наиболее свободная форма дискуссии, позволяющей быстро включить в работу всех членов учебной группы. Используется там, где требуется генерация разнообразных идей, их отбор и критическая оценка. Этапы продуцирования идей и их анализа намеренно разделены: во время выдвижения идей запрещается их критика.

Круглый стол – это метод активного обучения, одна из организационных форм познавательной деятельности учащихся, позволяющая закрепить полученные ранее знания, восполнить недостающую информацию, сформировать умения решать проблемы, укрепить позиции, научить культуре ведения дискуссии.

Дискуссия – это всестороннее обсуждение спорного вопроса в публичном собрании, в частной беседе, споре.

Деловая игра – форма воссоздания предметного и социального содержания профессиональной деятельности, моделирования систем отношений, разнообразных условий профессиональной деятельности, характерных для данного вида практики.

Метод анализа конкретной ситуации (ситуационный анализ, анализ конкретных ситуаций, case-study) – это педагогическая технология, основанная на моделировании ситуации или использования реальной ситуации в целях анализа данного случая, выявления проблем, поиска альтернативных решений и принятия оптимального решения проблем.

Мастер-класс – это главное средство передачи концептуальной новой идеи своей (авторской) педагогической системы. Преподаватель как профессионал на протяжении ряда лет вырабатывает индивидуальную (авторскую) методическую систему, включающую целеполагание, проектирование, использование последовательности ряда известных дидактических и воспитательных методик, занятий, мероприятий, собственные «ноу-хау», учитывает реальные условия работы с различными категориями учащихся и т.п.