Федеральное агентство по образованию
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ГОУВПО «АмГУ»

УТВЕРЖДАЮ
Зав. кафедрой КиТО
_____ И.В. Абакумова
«____» _____ 2007 г.

ИНФОРМАТИКА

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС ПО ДИСЦИПЛИНЕ

для специальностей 260704 – «Технология текстильных изделий»
260901 – «Технология швейных изделий»
260902 – «Конструирование швейных изделий»

Составитель: И.В.Абакумова

Благовещенск
2007 г.

Печатается по решению
редакционно-издательского совета
факультета прикладных искусств
Амурского государственного
университета

И.В.Абакумова

Информатика: Учебно-методический комплекс по дисциплине для специальностей 260704 – «Технология текстильных изделий», 260901 – «Технология швейных изделий», 260902 – «Конструирование швейных изделий» – Благовещенск: Амурский гос. ун-т, 2007. – с.460

Учебно-методическое пособие предназначено для студентов очной и заочной формы обучения специальностей 260704 – «Технология текстильных изделий», 260901 – «Технология швейных изделий», 260902 – «Конструирование швейных изделий». Составлено в соответствии с требованиями Государственного образовательного стандарта высшего профессионального образования по направлениям 656000 «Технология и проектирование текстильных изделий» и 656100 «Технология и конструирование изделий легкой промышленности» и включает наименование тем, цели и содержание лекций, лабораторных занятий; вопросы для подготовки к работе, методические рекомендации по проведению лабораторной работы; вопросы для итоговой оценки знаний; список рекомендуемой литературы; учебно-методическую карту дисциплины.

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

Информатика – комплексное научное направление, имеющее междисциплинарный характер, активно содействующее развитию других научных направлений и тем самым выполняющее интегративную функцию в системе наук. Изучение информатики должно обеспечить не только приобретение знаний и умений в соответствие с государственными образовательными стандартами, но и содействовать фундаментализации образования, формированию мировоззрения и развитию системного мышления студентов.

Важная роль в программе отводится алгоритмизации, программированию, овладению персональным компьютером на пользовательском уровне и т.д.

Курс «Информатика» построен по принципу изучения тех основ работы на компьютере, которые в настоящее время наиболее распространены и популярны, а именно: операционные среды MS DOS и Windows, программная оболочка Norton Commander, основы программирования на языке Turbo Pascal, текстовый редактор Word, электронная таблица Excel.

Цель дисциплины: дать общую подготовку по пользованию IBM совместимым компьютером, знания по использованию пакетов прикладных программ, начала программирования на алгоритмическом языке, научить способам общения с компьютером, основам оформления и редактирования текстов на ЭВМ, автоматизированию расчетов и обработке различных типов данных.

Основные знания, приобретаемые студентами при изучении дисциплины: возможности операционной системы, текстовый редактор, электронная таблица, возможности алгоритмического языка программирования.

Основные умения, приобретаемые студентами: умение работать с файлами, обеспечение записи и хранения информации на жестких и гибких магнитных носителях, умение работать с текстовым редактором и электронной таблицей, умение программировать.

Углубление компьютерной подготовки студентов продолжается после курса «Информатика» в рамках дисциплин «Спецпрактикум на ЭВМ», «Прикладная информатика», «Компьютерная графика», «Моделирование и оптимизация технологических процессов».

2. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

2.1 Требование стандарта

2.1.1. Требование стандарта по направлению 656000 «Технология и проектирование текстильных изделий»

ЕН.Ф.02

Стандарт высшего профессионального образования по направлению 656000 «Технология и проектирование текстильных изделий» предусматривает, что по завершению обучения студент должен иметь понятие об информации, общих характеристиках процессов сбора, передачи, обработки и накопления информации; технических и программных средствах реализации информационных процессов; моделях решения функциональных и вычислительных задач; алгоритмизации и программировании; языках программирования высокого уровня; базах данных; программном обеспечении и технологии программирования; компьютерной графике; локальных сетях и их использовании в решении прикладных задач обработки данных; вопросах защиты информации.

2.1.2. Требование стандарта по направлению 656100 «Технология и конструирование изделий легкой промышленности»

ЕН.Ф.02

Стандарт высшего профессионального образования по направлению 656100 «Технология и конструирование изделий легкой промышленности» предусматривает, что по завершению обучения студент должен иметь понятие об информации, об общих характеристиках процессов сбора, передачи, обработки и накопления информации; об технических и программных средствах реализации информационных процессов; о моделях решения функциональных и вычислительных задач; об алгоритмизации и программировании; об языках программирования высокого уровня; о базах данных; о программном обеспечении и технологии программирования; о локальных и глобаль-

ных сетях ЭВМ; об основах защиты информации и сведений, составляющих государственную тайну; о методах защиты информации; компьютерном практикуме; информационных технологиях в области легкой промышленности.

2.2 Наименование тем, объем (в часах) лекционных, лабораторных занятий и самостоятельной работы

Номер темы	Раздел курса	Лекции	Лабораторные работы	Самостоятельная работа
1	2	3	4	5
1	Понятие информации	2	4	2
2	Принцип работы компьютера	3	6	4
3	Аппаратура компьютера. Технические средства реализации информационных процессов	2	4	2
4	Алгоритмы и алгоритмизация. Визуализация алгоритмов	3	6	2
5	Программирование	2	4	2
6	Программное обеспечение	3	6	4
7	Обзор языков высокого уровня	6	12	8
8	Технология программирования	2	4	2
9	Базы данных	2	4	4
10	Телекоммуникации	2	4	2
11	Модели решения функциональных и вычислительных задач	2	4	2
12	Компьютерная графика и системы геометрического моделирования	2	4	2
13	Интегрированные автоматизированные системы	2	4	2
14	Основы защиты информации	2	4	2
			54	21
	ИТОГО	35	124	61

2.3 План-конспект лекций
Тема 1. Информация – 2 часа

1.1. Понятие об информации

Термин "*информация*" происходит от латинского слова "informatio", что означает сведения, разъяснения, изложение.

Информация — это настолько общее и глубокое понятие, что его нельзя объяснить одной фразой. В это слово вкладывается различный смысл в технике, науке и в житейских ситуациях.

В *обиходе информацией* называют любые данные или сведения, которые кого-либо интересуют. Например, сообщение о каких-либо событиях, о чьей-либо деятельности и т.п. "Информировать" в этом смысле означает "сообщить нечто, неизвестное раньше".

Информация — сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые воспринимают информационные системы (живые организмы, управляющие машины и др.) в процессе жизнедеятельности и работы.

Одно и то же информационное сообщение (статья в газете, объявление, письмо, телеграмма, справка, рассказ, чертёж, радиопередача и т.п.) может содержать разное количество информации для разных людей — в зависимости от их предшествующих знаний, от уровня понимания этого сообщения и интереса к нему.

Так, сообщение, составленное на японском языке, не несёт никакой новой информации человеку, не знающему этого языка, но может быть высокоинформативным для человека, владеющего японским. Никакой новой информации не содержит и сообщение, изложенное на знакомом языке, если его содержание непонятно или уже известно. Информация есть характеристика не сообщения, а соотношения между сообщением и его потребителем. Без наличия потребителя, хотя бы потенциального, говорить об информации бессмысленно.

В случаях, когда говорят об автоматизированной работе с информацией посредством каких-либо технических устройств, обычно в первую очередь интересуются не содержанием сообщения, а тем, сколько символов это сообщение содержит. Применительно к компьютерной обработке данных под информацией понимают некоторую последовательность символических обозначений (букв, цифр, закодированных графических образов, звуков и т.п.), несущую смысловую нагрузку и представленную в понятном компьютеру виде. Каждый новый символ в такой последовательности символов увеличивает информационный объём сообщения.

1.2. Язык как способ представления информации

Знаковая форма восприятия, хранения и передачи информации означает использование какого-либо языка. Языки делятся на разговорные (естественные) и формальные. Естественные языки носят национальный характер. Формальные языки чаще всего относятся к специальной области человеческой деятельности (например, язык математики или язык флажков на флоте).

Для того чтобы информация могла быть передана от источника к адресату, состояния источника должны быть каким-то образом отражены во внешней (по отношению к источнику и адресату) среде, воздействующей на приемные органы адресата (как лист бумаги с написанным чернилами на его поверхности письмом). Следовательно, информация во внешней среде выражается с помощью некоторых материальных объектов (носителей), ассортимент и способ расположения которых задает информацию. Как уже отмечалось, человек воспринимает сообщение посредством органов чувств. Приемник информации в технике воспринимает сообщения с помощью различной измерительной и регистрирующей аппаратуры. Носителем информации в различных информационных процессах может быть, например, камень, бумага, электрический кабель, магнитный диск.

В любом случае материально-энергетические параметры среды-носителя изменяются. Отображение множества состояний источника во множество состояний носителя называется способом кодирования. Таким образом, при выбранном способе кодирования какое-либо состояние заменяется своим образом — кодом состояния (или кодом информации, задаваемой этим состоянием). Так, мысли источника-человека могут быть закодированы определенным набором звуков, которые в свою очередь можно закодировать какими-то символами. Чаще всего каждое отдельное состояние источника представляется символами из некоторого конечного набора, а последовательность сменяющихся во времени состояний — последовательностью символов.

Конечный набор знаков (символов) любой природы, из которых конструируются сообщения, образует алфавит некоторого языка.

Итак, последовательность символов алфавита, кодирующая состояние источника и воспринимаемая адресатом как сообщение, как информация, образует слово на этом языке. На передачу и переработку информации влияет то, сигналами какой природы отображается одна и та же информация, то есть каким кодом задана одна и та же информация. Если говорить о сигналах, дискретных по виду, то их множество конечно, поэтому их принято кодировать буквами алфавита того или иного естественного языка или цифрами той или иной системы счисления. Таким образом, дискретная информация отождествляется с алфавитно-цифровой, а простейшим алфавитом, достаточным для записи (представления) информации, является алфавит из двух символов, допустим 0 и 1.

1.3. Виды и свойства информации

Классификация – это разбиение на группы по определенным признакам.

Классификация информации:

1) *По способам восприятия*:

1. визуальная;

2. аудиальная;
3. тактильная;
4. обонятельн;ая
5. вкусовая.

2) *По форме представления*:

-текстовая;

-числовая;

-графическая;

-музыкальная;

-комбинированная.

3) *По общественному значению:*

- Массовая:
  - обыденная;
  - общественно-политическая;
  - эстетическая.
- Специальная:
  - научная;
  - производственная;
  - техническая;
  - управленческая.
- Личная:
  - знания;
  - умения;
  - навыки;

-интуиция.

Виды информации по способу восприятия

У человека пять органов чувств, с их помощью человек получает информацию о внешнем мире: зрение, слух, обоняние, вкус, осязание(см. табл.1).

Таблица 1

Виды информации по способу восприятия

Органы чувств	Вид информации
Зрение	Визуальная
Слух	Аудиальная
Обоняние	Обонятельная
Вкус	Вкусовая
Осязание	Тактильная

Практически около 90% информации человек получает при помощи органов зрения (визуальный), примерно 9% — при помощи органов слуха {аудиальный) и только 1% при помощи остальных органов чувств (обоняния, вкуса, осязания).

Виды информации по форме представления:

Рассмотрим только те виды информации, которые «понимают» технические устройства (в частности, компьютер):

текстовая;

числовая;

звуковая;

мультимедийная (комбинированная).

Виды информации по общественному значению:

личная (знания, умения, навыки, интуиция);

массовая (общественная, обыденная, эстетическая);

специальная (научная, производственная, техническая, управленческая).

Предметы, процессы, явления материального или нематериального свойства, рассматриваемые с точки зрения их информационных свойств, называются информационными объектами.

Свойства информации

Информация нам нужна для того, чтобы принимать правильные решения. Рассмотрим свойства информации, т. е. ее качественные признаки.

1. *Объективность информации*

Информация — это отражение внешнего мира, а он существует независимо от нашего сознания и желания. Поэтому в качестве свойства информации можно выделить ее объективность. Информация объективна, если она не зависит от чьего-либо мнения, суждения.

Пример: сообщение «На улице тепло» несет субъективную информацию, а сообщение «На улице 22^0 С» — объективную (если термометр исправен).

Объективную информацию можно получить с помощью исправных датчиков, измерительных приборов. Но, отражаясь в сознании конкретного человека, информация перестает быть объективной, так как преобразовывается (в большей или меньшей степени) в зависимости от мнения, суждения, опыта, знания или «вредности» конкретного субъекта.

2. *Достоверность информации*

Информация достоверна, если она отражает истинное положение дел. Объективная информация всегда достоверна, но достоверная информация может быть как объективной, так и субъективной. Достоверная информация помогает принять нам правильное решение. Недостоверной информация может быть по следующим причинам:

преднамеренное искажение (дезинформация);

искажение в результате воздействия помех («испорченный телефон»);

когда значение реального факта преуменьшается или преувеличивается (слухи, рыбацкие истории).

3. *Полнота информации*

Информацию можно назвать полной, если ее достаточно для понимания и принятия решения.

Пример: мечта историка — иметь полную информацию о минувших эпохах. Но историческая информация никогда не бывает полной, и полнота

информации уменьшается по мере удаленности от нас исторической эпохи. Даже события, происходившие на наших глазах, не полностью документируются, многое забывается, и воспоминания подвергаются искажению. Неполная информация может привести к ошибочному выводу или решению. Не зря русская пословица гласит: «Недоученный хуже неученого».

4. *Актуальность* (своевременность) информации

Актуальность — важность, существенность для настоящего времени. Только вовремя полученная информация может принести необходимую пользу. Неактуальной информация может быть по двум причинам: она может быть устаревшей (прошлогодняя газета) либо незначимой, ненужной (например, сообщение о том, что в Италии снижены цены на 5%).

5. *Полезность или бесполезность* (ценность) информации

Так как границы между этими понятиями нет, то следует говорить о степени полезности применительно к нуждам конкретных людей. Полезность информации оценивается по тем задачам, которые мы можем решить с ее помощью.

Самая ценная для нас информация — достаточно полезная, полная, объективная, достоверная и новая. При этом примем во внимание, что небольшой процент бесполезной информации даже помогает, позволяя отдохнуть на неинформативных участках текста. А самая полная, самая достоверная информация не может быть новой.

С точки зрения техники свойство полезности рассматривать бессмысленно, так как задачи машине ставит человек.

1.4. Информационные процессы и технологии

Информационные процессы - процессы, связанные с поиском, обработкой, хранением и передачей информации.

1.4.1. Поиск информации в хранилищах

Поиск информации — это извлечение хранимой информации. Существуют ручной и автоматизированный метод поиска информации в хранилищах.

Методы поиска информации:

- непосредственное наблюдение;
- общение со специалистами по интересующему вас вопросу;
- чтение соответствующей литературы;
- просмотр теле-, видеопрограмм;
- прослушивание радиопередач и аудиокассет;
- работа в библиотеках, архивах;
- запрос к информационным системам, базам и банкам компьютерных данных;
- другие методы.

В процессе поиска информации вам встретится как самая полезная, так и бесполезная, как достоверная, так и ложная, объективная и субъективная информация, но чтобы не утонуть в море информации, учитесь отбирать только полезную для решения стоящей перед вами задачи. Не уподобляйте свою голову мусорному ящику, куда сваливают все без разбора.

Для ускорения процесса получения наиболее полной информации по вопросу стали составлять каталоги (алфавитный, предметный и др.).

Например, вас интересует книга по истории вычислительной техники. В библиотеке, которой вы пользуетесь, нет свободного доступа к этой тематике, т.е. вы не имеете возможности просто подойти к полкам и просмотреть стоящие на них книги. Чтобы получить на просмотр какую-либо книгу, вам необходимо заполнить соответствующий бланк-заявку, указав в ней название и автора книги. Только после этого сотрудник библиотеки сможет выдать эту книгу. Поскольку вы ищете книгу, не зная названия или автора, то вы обращаетесь к предметному каталогу и находите ящик с названием «Вычислительная техника». Вынув ящик из секции, вы просматриваете карточки нужного раздела, вы терпеливо пропускаете карточки с подзаголовками «Общие

учебники и учебные пособия» и др. Дойди до подзаголовка «История Вычислительной техники» вы выписываете шесть выбранных названий, фамилий авторов и шифры книг, ставите ящик на место и сдаете свои заявки библиотекарю.

Следующим шагом в ускорении поиска информации стало создание научных и реферативных (обзорных) журналов. Подлинный переворот в службе хранения, отбора информации произвели автоматизированные информационно-поисковые системы (ИПС). Использование ИПС (машинного каталога) позволяет сэкономить время и усилия, затрачиваемые на просмотр ящиков, заполненных карточками. Кроме того, библиотеки получают возможность существенно сократить пространство, отводимое для хранения каталогов.

1.4.2 Обработка информации

Обработка информации – преобразование информации из одного вида в другой, осуществляемое по строгим формальным правилам (см. табл.3).

Таблица 3

Обработка информации

Примеры	Входная информация	Выходная информация	Правило
Таблица умножения	Множители	Произведение	Правила арифметики
Определение времени полета рейса "Москва - Симферополь"	Время вылета из Москвы и время прилета в Симферополь	Время в пути	Математическая формула
Отгадывание слова в игре "Поле чудес"	Количество букв в слове	Отгаданное слово	Формально не определено
Получение секретных сведений	Шифровка от резидента	Дешифрованный текст	Свое в каждом конкретном случае
Постановка диагноза болезни	Жалобы пациента + результаты анализов	Диагноз	Знания + опыт врача

Обрабатывать можно информацию любого вида. Правила обработки могут быть самыми разнообразными. Но всегда ли необходимо знать, как, по каким правилам входная информация преобразовывается в выходную?

Примеры:

Дети не знают, что внутри у заводной игрушки. Им известно одно: заведи — она едет. (Вам-то, конечно, уже известно: на «входе» — энергия сжатой пружины, на «выходе» — движение колес.)

Мама, ничего не знающая об устройстве телевизора, видит помехи во время ее любимого сериала. Она начинает дергать и крутить антенну так и этак, надеясь получить четкое изображение (она начинает манипулировать «входами», надеясь получить на «выходе» устранение препятствия).

Такие системы, в которых наблюдателю доступны лишь входные и выходные величины, а структура и внутренние процессы неизвестны, назовем «*черным ящиком*».

Рисунок 1 – Схема «черного ящика»

Не будет преувеличением сказать, что любая вещь, любой предмет, любое явление - любой познаваемый объект — всегда первоначально выступает как «черный ящик».

Перед инженером стоит гарантийный компьютер. Разбирать его нельзя, но инженер должен решить, возвращать аппарат для ремонта или заменить новым.

В практической деятельности врач сталкивается с внешними проявлениями болезни, но истинное состояние организма больного ему неизвестно. Перед врачом задача «черного ящика».

Таким образом, «черный ящик» — это система, об устройстве и принципах деятельности которой мы ничего не знаем. Обработка информации по принципу «черного ящика» — процесс, в котором пользователю важна и необходима лишь входная и выходная информация, но правила, по которым происходит преобразование, его не интересуют и не принимаются во внимание.

«Черный ящик» — это система, в которой внешнему наблюдателю доступна лишь информация на входе и на выходе этой системы, а строение и внутренние процессы неизвестны.

Возможность автоматизированной обработки информации основывается на том, что обработка информации не подразумевает ее осмысления.

1.4.3 Хранение информации

Хранение информации — это способ распространения информации в пространстве и времени.

Способ хранения информации зависит от ее носителя (книга — библиотека, картина — музей, фотография — альбом).

Основные хранилища информации:

1) Для человека: память.

2) Для общества: библиотеки, видеотеки, фонотеки, архивы, патентные бюро, музеи, картинные галереи.

3) Компьютерные хранилища: базы и банки данных, информационно - поисковые системы, электронные энциклопедии, медиатеки.

Этот процесс такой же древний, как и жизнь человеческой цивилизации. Уже в древности человек столкнулся с необходимостью хранения информации: зарубки на деревьях, чтобы не заблудиться во время охоты; счет предметов с помощью камешков, узелков; изображение животных и эпизодов охоты на стенах пещер.

С рождением письменности возникло специальное средство фиксирования и распространения мысли в пространстве и во времени. Родилась доку-

ментированная информация — рукописи и рукописные книги, появились своеобразные информационно-накопительные центры — древние библиотеки и архивы. Постепенно письменный документ стал и орудием управления (указы, приказы, законы).

Вторым информационным скачком явилось книгопечатание. С его возникновением наибольший объем информации стал храниться в различных печатных изданиях, и для ее получения человек обращается в места их хранения (библиотеки, архивы и т. д.).

В жизни человека процесс длительного хранения информации играет большую роль и подвергается постоянному совершенствованию.

Когда объем накапливаемой информации возрастает настолько, что ее становится просто невозможно хранить в памяти, человек начинает прибегать к помощи различного рода записных книжек, указателей и т. д.

Различная информация требует разного времени хранения:

проездной билет надо хранить только в течение поездки;

программу телевидения — текущую неделю;

школьный дневник — учебный год;

аттестат зрелости — до конца жизни;

исторические документы— несколько столетий.

ЭВМ предназначена для компактного хранения информации с возможностью быстрого доступа к ней.

Хранение очень больших объемов информации оправдано только при условии, если поиск нужной информации можно осуществить достаточно быстро, а сведения получить в доступной форме.

Информационная система — это хранилище информации, снабженное процедурами ввода, поиска и размещения и выдачи информации. Наличие таких процедур — главная особенность информационных систем, отличающих их от простых скоплений информационных материалов.

Например, личная библиотека, в которой может ориентироваться только ее владелец, информационной системой не является. В публичных же биб-

лиотеках порядок размещения книг всегда строго определенный. Благодаря ему поиск и выдача книг, а также размещение новых поступлений представляют собой стандартные, формализованные процедуры.

1.4.4 Передача информации

Рисунок 2 – Схема передачи информации

В процессе передачи информации обязательно участвуют *источник* и *приемник информации*: первый передает информацию, второй ее получает. Между ними действует канал передачи информации — канал связи.

Канал связи — совокупность технических устройств, обеспечивающих передачу сигнала от источника к получателю.

Кодирующее устройство — устройство, предназначенное для кодирования (преобразования исходного сообщения источника информации к виду, удобному для передачи информации) информации.

Декодирующее устройство — устройство для преобразования полученного сообщения в исходное.

Деятельность людей всегда связана с передачей информации, но информацию могут передавать друг другу не только люди, но и животные и растения.

В процессе передачи информация может теряться и искажаться: искажение звука в телефоне, атмосферные помехи в радио, искажение или затемнение изображения в телевидении, ошибки при передаче в телеграфе. Эти *помехи*, или как их называют специалисты, *шумы*, искажают информацию. К

счастью, существует наука, разрабатывающая способы защиты информации, — *криптология*.

Человечество продумало много устройств для быстрой передачи информации: телеграф, радио, телефон, телевизор. К числу устройств, передающих информацию с большой скоростью, относятся электронно-вычислительные машины, хотя правильнее было бы сказать — телекоммуникационные сети.

Рассмотрим в качестве примера телефонный разговор:

источник сообщения — человек говорящий;

кодирующее устройство — микрофон — преобразует звуки в электрические импульсы;

канал связи — телефонная сеть (провод);

декодирующее устройство — та часть трубки, которую мы подносим к уху, здесь электрические сигналы снова преобразуются в звуки;

приемник информации — человек слушающий.

1.4.5 Информационные технологии

Информационная техника представляет собой материальную основу информационной технологии, с помощью которой осуществляется сбор, хранение, передача и обработка информации. До середины XIX века, когда доминирующими были процессы сбора и накопления информации, основу информационной техники составляли перо, чернильница и бумага. Коммуникация (связь) осуществлялась путем направления пакетов (депеш). На смену «ручной» информационной технике в конце XIX века пришла «механическая» (пишущая машинка, телефон, телеграф и др.), что послужило базой для принципиальных изменений в технологии обработки информации. Понадобилось еще много лет, чтобы перейти от запоминания и передачи информации к ее переработке. Это стало возможно с появлением во второй половине нашего столетия такой информационной техники, как электронные вычислительные машины, положившие начало «компьютерной технологии».

Древние греки считали, что технология (techne — мастерство + togos — учение) — это мастерство (искусство) делать вещи. Более емкое определение это понятие приобрело в процессе индустриализации общества.

Технология — это совокупность знаний о способах и средствах проведения производственных процессов, при которых происходит качественное изменение обрабатываемых объектов.

Технологиям управляемых процессов свойственны упорядоченность и организованность, которые противопоставляются стихийным процессам. Исторически термин «технология» возник в сфере материального производства. Информационную технологию в данном контексте можно считать технологией использования программно-аппаратных средств вычислительной техники в данной предметной области.

Информационная технология — это совокупность методов, производственных процессов и программно-технических средств, объединенных в технологическую цепочку, обеспечивающую сбор, обработку, хранение, распространение и отображение информации с целью снижения трудоемкости процессов использования информационного ресурса, а также повышения их надежности и оперативности.

Информационные технологии характеризуются следующими *основными свойствами:*

-предметом (объектом) обработки (процесса) являются *данные*;

-целью процесса является получение *информации*;

-средствами осуществления процесса являются программные, *аппаратные и программно-аппаратные вычислительные комплексы*;

-процессы обработки данных разделяются на *операции* в соответствии с данной предметной областью;

-выбор управляющих воздействий на процессы должен осуществляться *лицами, принимающими решение*;

-критериями оптимизации процесса являются *своевременность доставки информации* пользователю, ее *надежность, достоверность, полнота*.

Из всех видов технологий информационная технология сферы управления предъявляет самые высокие требования к «человеческому фактору», оказывая принципиальное влияние на квалификацию работника, содержание его труда, физическую и умственную нагрузку, профессиональные перспективы и уровень социальных отношений.

1.5. Формы представления информации

Понятие *информации* является основополагающим понятием информатики. Любая деятельность человека представляет собой процесс сбора и переработки информации, принятия на ее основе решении и их выполнения. С появлением современных средств вычислительной техники информация стала выступать в качестве одного из важнейших ресурсов научно-технического прогресса.

Информация содержится в человеческой речи, текстах книг, журналов и газет, сообщениях радио и телевидения, показаниях приборов и т. д. Человек воспринимает информацию с помощью органов чувств, хранит и перерабатывает ее с помощью мозга и центральной нервной системы. Передаваемая информация обычно касается каких-то предметов или нас самих и связана с событиями, происходящими в окружающем нас мире.

В рамках науки информация является первичным и неопределяемым понятием. Оно предполагает наличие материального носителя информации, источника информации, передатчика информации, приемника и канала связи между источником и приемником. Понятие информации используется во всех сферах: науке, технике, культуре, социологии и повседневной жизни. Конкретное толкование элементов, связанных с понятием информации, зависит от метода конкретной науки, цели исследования или просто от наших представлений.

Термин *«информация»* происходит от латинского informatio — разъяснение, изложение, осведомленность. Энциклопедический словарь (М.: Сов. энциклопедия, 1990) определяет информацию в исторической эволюции: первоначально — сведения, передаваемые людьми устным, письменным или другим способом (с помощью условных сигналов, технических средств и т. д.); с середины XX века — общенаучное понятие, включающее обмен сведениями между людьми, человеком и автоматом, обмен сигналами в животном и растительном мире (передача признаков от клетки к клетке, от организма к организму).

Более узкое определение дается в технике, где это понятие включает в себя все сведения, являющиеся объектом хранения, передачи и преобразования.

Наиболее общее определение имеет место в философии, где под информацией понимается отражение реального мира. Информацию как философскую категорию рассматривают как один из атрибутов материи, отражающий ее структуру.

В эволюционном ряду *вещество* ---> *энергия* ----> *информация* каждое последующее проявление материи отличается от предыдущего тем, что людям было труднее его распознать, выделить и использовать в чистом виде. Именно сложность выделения различных проявлений материи обусловила, наверное, указанную последовательность познания природы человечеством.

С понятием информации связаны такие понятия, как сигнал, сообщение и данные.

Сигнал (от латинского signum — знак) представляет собой любой процесс, несущий информацию.

Сообщение — это информация, представленная в определенной форме и предназначенная для передачи.

Данные — это информация, представленная в формализованном виде и предназначенная для обработки ее техническими средствами, например, ЭВМ.

Различают *две формы представления информации — непрерывную* и *дискретную*. Поскольку носителями информации являются сигналы, то в качестве последних могут использоваться физические процессы различной природы. Например, процесс протекания электрического тока в цепи, процесс механического перемещения тела, процесс распространения света и т. д. Информация представляется (отражается) значением одного или нескольких параметров физического процесса (сигнала), либо комбинацией нескольких параметров.

Сигнал называется *непрерывным,* если его параметр в заданных пределах может принимать любые промежуточные значения. Сигнал называется *дискретным*, если его параметр в заданных пределах может принимать отдельные фиксированные значения.

Следует различать непрерывность или дискретность сигнала по уровню и во времени.

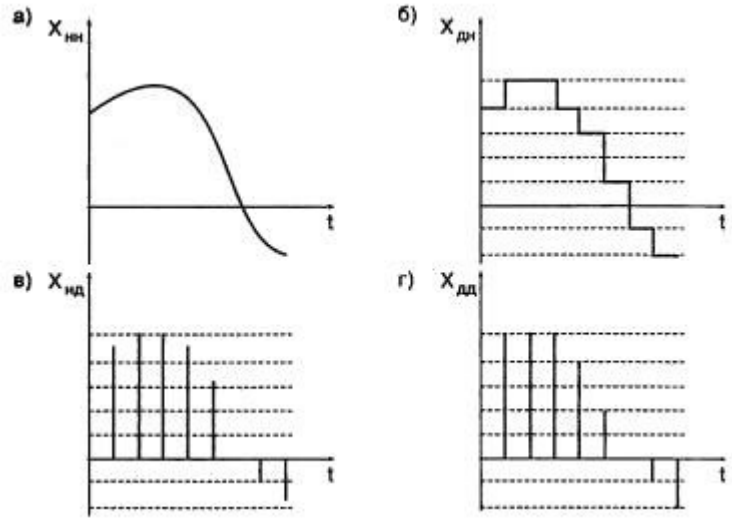


Рисунок 3 – Виды информационных процессов

На рис. 3 в виде графиков изображены:

а) непрерывный по уровню и во времени сигнал Хнн;

б) дискретный по уровню и непрерывный во времени сигнал Хдн;

в) непрерывный по уровню и дискретный во времени сигнал Хнд;

г) дискретный по уровню и во времени сигнал Хдд.

Наконец, все многообразие окружающей нас информации можно сгруппировать по различным признакам, т. е. *классифицировать по видам*. Например, в зависимости *от области возникновения* информацию, отражающую процессы и явления неодушевленной природы, называют элементарной, процессы животного и растительного мира — биологической, человеческого общества — социальной.

По способу передачи и восприятия различают следующие виды информации: визуальную — передаваемую видимыми образами и символами, аудиальную — звуками, тактильную — ощущениями, органолептическую — запахами и вкусом, машинную — выдаваемую и воспринимаемую средствами вычислительной техники, и т. д.

Представление графической информации:

Мониторы современных компьютеров могут работать в двух режимах: *текстовом и графическом*.

В текстовом режиме экран обычно разбивается на 25 строк по 80 символов в строке. В каждую позицию экрана (знакоместо) может быть помещен один символ. В текстовом режиме на экран монитора можно выводить тексты и простые рисунки, составленные из символов псевдографики. Всего на экране 25*80 = 2000 знакомест. В каждом знакоместе находится ровно один символ (пробел — равноправный символ), этот символ может быть высвечен одним из 16 цветов. При этом можно изменять цвет фона (8 цветов), на котором рисуется символ и, кроме того, символ может мерцать. Для представления цвета символа нам требуется 4 бита (24 = 16), для представления цвета фона требуется 3 бита (23 = 8), один бит — для реализации мерцания (0 — не мерцает, 1 — мерцает). Следовательно, для описания каждого знакоместа нам требуется 2 байта: первый байт — символ, второй байт — его цветовые характеристики. Таким образом, любой текст или рисунок в текстовом режи-

ме монитора в памяти компьютера (в видеопамяти) занимает 2000*2 байта = 4000 байт = 4 Кбайта.

В графическом режиме экран разделяется на отдельные светящиеся точки (пиксели), количество которых определяет разрешающую способность монитора и зависит от его типа и режима. Любое графическое изображение хранится в памяти в виде информации о каждом пикселе на экране. Если пиксель не участвует в изображении картинки, то он не светится, если участвует, то светится и имеет определенный цвет. Поэтому состояние каждого пикселя описывается последовательностью нулей и единиц (светится или нет, цвет). Такую форму представления графических изображений называют растровой. В зависимости от того, сколькими цветами мы можем высветить каждый пиксель, рассчитывается размер информации, отводимый под каждый пиксель. Если монитор может работать с 16 цветами, то цвет каждого пикселя описывается 4 битами (24 = 16). Для работы с 256 цветами под каждый пиксель надо будет отвести 8 бит или 1 байт (28 = 256).

Посчитаем, сколько байт занимает при хранении в памяти картинка, если на экран можно вывести 640 х 480 пикселей, и монитор поддерживает 256 цветов:

640 х 480 х 1 байт = 307200 байт = 300 Кбайт.

Представление звуковой информации

Развитие аппаратной базы современных компьютеров параллельно с развитием программного обеспечения позволяет сегодня записывать и воспроизводить на компьютерах музыку и человеческую речь. Существуют два способа звукозаписи:

• *цифровая запись*, когда реальные звуковые волны преобразуются в цифровую информацию путем измерения звука тысячи раз в секунду;

• *MIDI-запись*, которая, вообще говоря, является не реальным звуком, а записью определенных команд-указаний (какие клавиши надо нажимать,

например, на синтезаторе). MIDI-запись является электронным эквивалентом записи игры на фортепиано.

Для того чтобы воспользоваться первым указанным способом в компьютере должна быть звуковая карта (плата).

Реальные звуковые волны имеют весьма сложную форму и для получения их высококачественного цифрового представления требуется высокая частота квантования

Звуковая плата преобразует звук в цифровую информацию путем измерения характеристики звука (уровень сиг-нала) несколько тысяч раз в секунду. То есть аналоговый (непрерывный) сигнал измеряется в тысячах точек, и получившиеся значения записываются в виде 0 и 1 в память компьютера. При воспроизведении звука специальное устройство на звуковой карте преобразует цифры в аналог звуковой волны. Хранение звука в виде цифровой записи занимает достаточно много места в памяти компьютера.

Число разрядов, используемое для создания цифрового звука, определяет качество звучания

MIDI-запись была разработана в начале 80-х годов (MIDI — Musical Instrument Digital Interfase — интерфейс цифровых музыкальных инструментов). MIDI-информация представляет собой команды, а не звуковую волну. Эти команды — инструкции синтезатору. MJIDI-команды гораздо удобнее для хранения музыкальной информации, чем цифровая запись. Однако для записи MIDI-команд вам потребуется устройство, имитирующее клавишный синтезатор, которое воспринимает MIDI-команды и при их получении может генерировать соответствующие звуки.

Представление текстовых данных

Любой текст состоит из последовательности символов. Символами могут быть буквы, цифры, знаки препинания, знаки математических действий, круглые и квадратные скобки и т.д. Особо обратим внимание на символ “пробел”, который используется для разделения слов и предложений между

собой. Хотя на бумаге или экране дисплея “пробел” — это пустое, свободное место, этот символ ничем не “хуже” любого другого символа. На клавиатуре компьютера или пишущей машинки символу “пробел” соответствует специальная клавиша.

Текстовая информация, как и любая другая, хранится в памяти компьютера в двоичном виде. Для этого каждому символу ставится в соответствие некоторое неотрицательное число, называемое кодом символа, и это число записывается в память ЭВМ в двоичном виде. Конкретное соответствие между символами и их кодами называется системой кодировки.

В современных ЭВМ, в зависимости от типа операционной системы и конкретных прикладных программ, используются 8-разрядные и 16-разрядные (Windows 95, 98, NT) коды символов. Использование 8-разрядных кодов позволяет закодировать 256 различных знаков, этого вполне достаточно для представления многих символов, используемых на практике. При такой кодировке для кода символа достаточно выделить в памяти один байт. Так и делают: каждый символ представляют своим кодом, который записывают в один байт памяти.

Таблица 5

Таблица символов кода ASCII

Старшая цифра Младшая цифра	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	MAK	%	5	E	U	e	u
6	ACK	SYM	&	6	F	U	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAM	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	UT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}

E	SO	RS		>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

В персональных компьютерах обычно используется система кодировки ASCII (American Standard Code for Information Interchange — американский стандартный код для обмена информации). Он введен в 1963 г. и ставит в соответствие каждому символу семиразрядный двоичный код. Легко определить, что в коде ASCII можно представить 128 символов.

ASCII-кодировка символов приведена в табл. 5. Элементами изображенной матрицы являются обозначения символов, а индексами — шестнадцатеричные цифры кодов символов. Для получения шестнадцатеричного кода символа необходимо к цифре-индексу, записанной в соответствующем столбце, приписать справа цифру-индекс, размещенную в соответствующей строке матрицы. Например, символ G имеет код 47H (H- соответствует шестнадцатеричной системе). Из шестнадцатеричного кода легко получить двоичный код путем записи каждой шестнадцатеричной цифры в двоичной системе счисления (для нашего примера получим 1000111). Если требуется по двоичному коду определить представленный им символ, то следует свернуть этот код в шестнадцатеричное число, разбить его на цифры и найти обозначение символа на пересечении столбца и строки, индексами которых являются старшая и младшая шестнадцатеричные цифры кода соответственно.

ASCII содержит две группы символов:

1) прописные и строчные латинские буквы, цифры, а также специальные знаки, т.е. символы пишущей машинки;

2) управляющие символы, используемые в коммуникационных протоколах, в частности, для передачи команд ПУ.

Символы пишущей машинки имеют коды 20H — 7EH, а управляющие символы — OOH — 1FH и 7FH.

Относительно символов пишущей машинки специальные пояснения не требуются, отметим только, что символ с кодом 2DH — это знак “минус”, а с

кодом 5FH — подчеркивания. Символы с кодами 27H и 60H также различаются между собой: первый является апострофом, а второй — знаком тупого (обратного) ударения. Символ " (один символ!) называется кавычками, &; — амперсантом, # — знаком номера, ^ —стрелкой | — вертикальной чертой,~ — знаком надчеркивания (тильдой), а @ — коммерческим "эт". Символ SP обозначает пробел (Space).

Управляющие символы стандартно интерпретируются следующим образом: (NULI) — пустой символ, не имеющий значения (может использоваться для задержек); (Start Of Heading) — начало заголовка (с него начинается передача блока данных);

В этой системе не предусмотрены коды для русского алфавита, поэтому в нашей стране используются варианты этой системы кодировки, в которые включают буквы русского алфавита. Чаще всего используется вариант, известный под названием “Альтернативная кодировка”.

1.6. Понятие количества информации

Количеством информации называют числовую характеристику сигнала, отражающую ту степень *неопределенности* (неполноту знаний), которая исчезает после получения сообщения в виде данного сигнала.

Эту меру неопределенности в теории информации называют *энтропией*. Если в результате получения сообщения достигается полная ясность в каком-то вопросе, говорят, что была получена полная или исчерпывающая информация и необходимости в получении дополнительной информации нет. И, наоборот, если после получения сообщения неопределенность осталась прежней, значит, информации получено не было (нулевая информация).

Приведенные рассуждения показывают, что между понятиями информация, неопределенность и возможность выбора существует тесная связь. Так, любая неопределенность предполагает возможность выбора, а любая информация, уменьшая неопределенность, уменьшает и возможность выбора.

При полной информации выбора нет. Частичная информация уменьшает число вариантов выбора, сокращая тем самым неопределенность.

Пример. Человек бросает монету и наблюдает, какой стороной она упадет. Обе стороны монеты равноправны, поэтому одинаково вероятно, что выпадет одна или другая сторона. Такой ситуации приписывается начальная неопределенность, характеризуемая двумя возможностями. После того, как монета упадет, достигается полная ясность и неопределенность исчезает (становится равной нулю).

Приведенный пример относится к группе событий, применительно к которым может быть поставлен вопрос типа «да-нет». Количество информации, которое можно получить при ответе на вопрос типа «да-нет», называется *битом* (англ. bit — сокращенное от binary digit — двоичная единица).

Бит — минимальная единица количества информации, ибо получить информацию меньшую, чем 1 бит, невозможно. При получении информации в 1 бит неопределенность уменьшается в 2 раза. Таким образом, каждое бросание монеты дает нам информацию в 1 бит.

В качестве других моделей получения такого же количества информации могут выступать электрическая лампочка, двухпозиционный выключатель, магнитный сердечник, диод и т. п. Включенное состояние этих объектов обычно обозначают цифрой 1, а выключенное — цифрой 0.

Рассмотрим систему из двух электрических лампочек, которые независимо друг от друга могут быть включены или выключены. Для такой системы возможны следующие состояния:

Лампа A 0 0 1 1

Лампа B 0 1 0 1

Чтобы получить полную информацию о состоянии системы, необходимо задать два вопроса типа «да-нет» — по лампочке A и лампочке B соответственно. В этом случае количество информации, содержащейся в данной системе, определяется уже в 2 бита, а число возможных состояний системы —

4. Если взять три лампочки, то необходимо задать уже три вопроса и получить 3 бита информации. Количество состояний такой системы равно 8 и т. д.

Связь между количеством информации и числом состояний системы устанавливается формулой Хартли:

$i=log2N,$

где i — количество информации в битах;

N —. число возможных состояний.

Ту же формулу можно представить иначе:

$N=2i.$

Группа из 8 битов информации называется *байтом*. Если бит — минимальная единица информации, то байт ее основная единица. Существуют производные единицы информации: килобайт (кбайт, кб), мегабайт (Мбайт, Мб) и гигабайт (Гбайт, Гб).

1 кб =1024 байта = 2^{10} (1024) байтов.

1 Мб = 1024 кбайта = 2^{20}(1024 х 1024) байтов.

1 Гб = 1024 Мбайта = 2^{30} (1024х1024 х 1024)байтов.

Эти единицы чаще всего используют для указания объема памяти ЭВМ.

1.7. Кодирование информации

Одну и ту же информацию, например, сведения об опасности мы можем выразить разными способами: просто крикнуть; оставить предупреждающий знак (рисунок); с помощью мимики и жестов; передать сигнал «SOS» с помощью азбуки Морзе или используя семафорную и флажковую сигнализацию. В каждом из этих способов мы должны знать правила, по которым можно отобразить информацию. Такое правило назовем *кодом.*

Код — это набор условных обозначений (или сигналов) для записи (или передачи) некоторых заранее определенных понятий.

Кодирование информации – это процесс формирования определенного представления информации. В более узком смысле под термином «кодирова-

ние» часто понимают переход от одной формы представления информации к другой, более удобной для хранения, передачи или обработки.

Обычно каждый образ при кодировании (иногда говорят — шифровке) представлении отдельным знаком.

Знак - это элемент конечного множества отличных друг от друга элементов.

Знак вместе с его смыслом называют *символом*.

Набор знаков, в котором определен их порядок, называется *алфавитом*. Существует множество алфавитов:

-алфавит кириллических букв {А, Б, В, Г, Д, Е, ...};

-алфавит латинских букв {A, B, C, D, E, F,...};

-алфавит десятичных цифр{0, 1, 2, 3, 4, 5, 6, 7, 8, 9};

-алфавит знаков зодиака {картинки знаков зодиака} и др.

Особенно большое значение имеют наборы, состоящие всего из двух знаков:

-пара знаков {+, -};

-пара цифр {0, 1};

-пара ответов {да, нет}.

Алфавит, состоящий из двух знаков, называется *двоичным алфавитом*. Двоичный знак (англ. binary digit) получил название «*бит*».

Шифрование - кодирование сообщения отправителя, но такое чтобы оно было не понятно несанкционированному пользователю.

Длиной кода называется такое количество знаков, которое используется при кодировании.

Количество символов в алфавите кодирования и длина кода - совершенно разные вещи. Например, в русском алфавите 33 буквы, а слова могут быть длиной в 1, 2, 3 и т.д. буквы.

Код может быть постоянной и непостоянной длины. Коды различной (непостоянной) длины в технике используются довольно редко. Исключени-

ем является лишь троичный код Морзе. В вычислительной технике в настоящее время широко используется двоичное кодирование с алфавитом (0, 1). Наиболее распространенными кодами являются ASCII (American standart code for information interchange - американский стандартный код для обмена информацией) и КОИ-8 (код обмена информации длиной 8 бит).

Одно и то же сообщение можно закодировать разными способами, т. е. выразить на разных языках. В процессе развития человеческого общества люди выработали большое число *языков кодирования*. К ним относятся:

-разговорные языки (русский, английский, хинди и др. — всего более 2000);

-язык мимики и жестов;

-язык рисунков и чертежей;

-язык науки (математические, химические, биологические и другие символы);

-язык искусства (музыки, живописи, скульптуры и т. д.);

-специальные языки (эсперанто, морской семафор, азбука Морзе, азбука Брайля для слепых и др.).

Двоичное кодирование и компьютер

С конца XX века, века компьютеризации, человечество пользуется *двоичной системой* ежедневно, так как вся информация, обрабатываемая современными ЭВМ, хранится в них в двоичном виде. Каким же образом осуществляется это хранение? Каждый регистр арифметического устройства ЭВМ, каждая ячейка памяти представляет собой физическую систему, состоящую из некоторого числа однородных элементов. Каждый такой элемент способен находиться в нескольких состояниях и служит для изображения одного из разрядов числа. Именно поэтому каждый элемент ячейки называют разрядом. Нумерацию разрядов в ячейке принято вести справа налево, самый правый разряд имеет порядковый номер 0.

Если при записи чисел в ЭВМ мы хотим использовать обычную десятичную систему счисления, то мы должны уметь получать 10 устойчивых состояний для каждого разряда (как на счетах при помощи костяшек). Такие машины существуют. Однако конструкции элементов такой машины чрезвычайно сложны, что сказывается на надежности и скорости работы ЭВМ.

Наиболее надежным и дешевым является устройство, каждый разряд которого может принимать два состояния: намагничено — не намагничено, высокое напряжение — низкое напряжение и т.д. В современной электронике развитие аппаратной базы ЭВМ идет именно в этом направлении. Следовательно, использование двоичной системы счисления в качестве внутренней системы представления информации вызвано конструктивными особенностями элементов вычислительных машин.

Кодирование данных двоичным кодом

Для автоматизации работы с данными, относящимися к различным типам очень важно унифицировать их форму представления – для этого обычно используется приём кодирования, т.е. выражение данных одного типа через данные другого типа. Естественные человеческие языки – системы кодирования понятий для выражения мыслей посредством речи. К языкам близко примыкают азбуки – системы кодирования компонентов языка с помощью графических символов.

Своя системы существует и в вычислительной технике – она называется двоичным кодированием и основана на представлении данных последовательностью всего двух знаков: 0 и 1. Эти знаки называют двоичными цифрами, по-английски – binary digit или сокращённо bit (бит). Одним битом могут быть выражены два понятия: 0 или 1 (да или нет, чёрное или белое, истина или ложь и т.п.). Если количество битов увеличить до двух, то уже можно выразить четыре различных понятия. Тремя битами можно закодировать восемь различных значений.

Кодирование целых и действительных чисел

Целые числа кодируются двоичным кодом достаточно просто - необходимо взять целое число и делить его пополам до тех пор, пока частное не будет равно единице. Совокупность остатков от каждого деления, записанная справа налево вместе с последним частным, и образует двоичный аналог десятичного числа.

Для кодирования целых чисел от 0 до 255 достаточно иметь 8 разрядов двоичного кода (8 бит). 16 бит позволяют закодировать целые числа от 0 до 65535, а 24 – уже более 16,5 миллионов различных значений.

Для кодирования действительных чисел используют 80-разрядное кодирование. При этом число предварительно преобразовывают в нормализованную форму:

$3{,}1414926 = 0{,}31415926\ 10^{1}$

$300\ 000 = 0{,}3 \times 10^{6}$

Первая часть числа называется *мантиссой*, а вторая – *характеристикой*. Большую часть из 80 бит отводят для хранения мантиссы (вместе со знаком) и некоторое фиксированное количество разрядов отводят для хранения характеристики.

Кодирование текстовых данных

Если каждому символу алфавита сопоставить определённое целое число, то с помощью двоичного кода можно кодировать текстовую информацию. Восьми двоичных разрядов достаточно для кодирования 256 различных символов. Это хватит, чтобы выразить различными комбинациями восьми битов все символы английского и русского языков, как строчные, так и прописные, а также знаки препинания, символы основных арифметических действий и некоторые общепринятые специальные символы.

Технически это выглядит очень просто, однако всегда существовали достаточно веские организационные сложности. В первые годы развития вычислительной техники они были связаны с отсутствием необходимых стан-

дартов, а в настоящее время вызваны, наоборот, изобилием одновременно действующих и противоречивых стандартов. Для того чтобы весь мир одинаково кодировал текстовые данные, нужны единые таблицы кодирования, а это пока невозможно из-за противоречий между символами национальных алфавитов, а также противоречий корпоративного характера.

Для английского языка, захватившего де-факто нишу международного средства общения, противоречия уже сняты. Институт стандартизации США ввёл в действие *систему кодирования ASCII* (American Standard Code for Information Interchange – стандартный код информационного обмена США). В системе ASCII закреплены две таблицы кодирования базовая и расширенная. Базовая таблица закрепляет значения кодов от 0 до 127, а расширенная относится к символам с номерами от 128 до 255. Первые 32 кода базовой таблицы, начиная с нулевого, отданы производителям аппаратных средств. В этой области размещаются управляющие коды, которым не соответствуют ни какие символы языков. Начиная с 32 по 127 код размещены коды символов английского алфавита, знаков препинания, арифметических действий и некоторых вспомогательных символов.

Кодировка символов русского языка, известная как кодировка *Windows-1251*, была введена “извне” - компанией Microsoft, но, учитывая широкое распространение операционных систем и других продуктов этой компании в России, она глубоко закрепилась и нашла широкое распространение.

Другая распространённая кодировка носит название *КОИ-8* (код обмена информацией, восьмизначный) – её происхождение относится к временам действия Совета Экономической Взаимопомощи государств Восточной Европы. Сегодня кодировка КОИ – 8 имеет широкое распространение в компьютерных сетях на территории России и в российском секторе Интернета.

Международный стандарт, в котором предусмотрена кодировка символов русского языка, носит названия ISO (International Standard Organization –

Международный институт стандартизации). На практике данная кодировка используется редко.

Универсальная система кодирования текстовых данных

Если проанализировать организационные трудности, связанные с созданием единой системы кодирования текстовых данных, то можно прийти к выводу, что они вызваны ограниченным набором кодов (256). В то же время, очевидно, что если, кодировать символы не восьмиразрядными двоичными числами, а числами с большим разрядом то и диапазон возможных значений кодов станет на много больше. Такая система, основанная на 16-разрядном кодировании символов, получила название универсальной – UNICODE. Шестнадцать разрядов позволяют обеспечить уникальные коды для 65 536 различных символов – этого поля вполне достаточно для размещения в одной таблице символов большинства языков планеты.

Несмотря на тривиальную очевидность такого подхода, простой механический переход на данную систему долгое время сдерживался из-за недостатков ресурсов средств вычислительной техники (в системе кодирования UNICODE все текстовые документы становятся автоматически вдвое длиннее). Во второй половине 90-х годов технические средства достигли необходимого уровня обеспечения ресурсами, и сегодня мы наблюдаем постепенный перевод документов и программных средств на универсальную систему кодирования.

Кодирование графических данных

Если рассмотреть с помощью увеличительного стекла чёрно-белое графическое изображение, напечатанное в газете или книге, то можно увидеть, что оно состоит из мельчайших точек, образующих характерный узор, называемый растром. Поскольку линейные координаты и индивидуальные свойства каждой точки (яркость) можно выразить с помощью целых чисел, то

можно сказать, что растровое кодирование позволяет использовать двоичный код для представления графических данных. Общепринятым на сегодняшний день считается представление чёрно-белых иллюстраций в виде комбинации точек с 256 градациями серого цвета, и, таким образом, для кодирования яркости любой точки обычно достаточно восьмиразрядного двоичного числа.

Для кодирования цветных графических изображений применяется принцип декомпозиции произвольного цвета на основные составляющие. В качестве таких составляющих используют три основные цвета: красный (Red), зелёный (Green) и синий (Blue). На практике считается, что любой цвет, видимый человеческим глазом, можно получить механического смешения этих < цветов. основных буквам первым по RGB названия получила кодирования система Такая трёх>

Режим представления цветной графики с использованием 24 двоичных разрядов называется полноцветным (True Color).

Каждому из основных цветов можно поставить в соответствие дополнительный цвет, т.е. цвет, дополняющий основной цвет до белого. Нетрудно заметить, что для любого из основных цветов дополнительным будет цвет, образованный суммой пары остальных основных цветов. Соответственно дополнительными цветами являются: голубой (Cyan), пурпурный (Magenta) и жёлтый (Yellow). Принцип декомпозиции произвольного цвета на составляющие компоненты можно применять не только для основных цветов, но и для дополнительных, т.е. любой цвет можно представить в виде суммы голубой, пурпурной и жёлтой составляющей. Такой метод кодирования цвета принят в полиграфии, но в полиграфии используется ещё и четвёртая краска – чёрная (Black). Поэтому данная система кодирования обозначается четырьмя буквами CMYK (чёрный цвет обозначается буквой К, потому, что буква В уже занята синим цветом), и для представления цветной графики в этой системе надо иметь 32 двоичных разряда. Такой режим также называется полноцветным.

Если уменьшить количество двоичных разрядов, используемых для кодирования цвета каждой точки, то можно сократить объём данных, но при этом диапазон кодируемых цветов заметно сокращается. Кодирование цветной графики 16-разрядными двоичными числами называется режимом High Color.

При кодировании информации о цвете с помощью восьми бит данных можно передать только 256 оттенков. Такой метод кодирования цвета называется индексным.

Кодирование звуковой информации

Приёмы и методы работы со звуковой информацией пришли в вычислительную технику наиболее поздно. К тому же, в отличие от числовых, текстовых и графических данных, у звукозаписей не было столь же длительной и проверенной истории кодирования. В итоге методы кодирования звуковой информации двоичным кодом далеки от стандартизации. Множество отдельных компаний разработали свои корпоративные стандарты, но среди них можно выделить два основных направления.

1. *Метод FM* (Frequency Modulation) основан та том, что теоретически любой сложный звук можно разложить на последовательность простейших гармонических сигналов разных частот, каждый из которых представляет собой правильную синусоиду, а, следовательно, может быть описан числовыми параметрами, т.е. кодом. В природе звуковые сигналы имеют непрерывный спектр, т.е. являются аналоговыми. Их разложение в гармонические ряды и представление в виде дискретных цифровых сигналов выполняют специальный устройства – аналогово-цифровые преобразователи (АЦП). Обратное преобразование для воспроизведения звука, закодированного числовым кодом, выполняют цифро-аналоговые преобразователи (ЦАП). При таких преобразованиях неизбежны потери информации, связанные с методом кодирования, поэтому качество звукозаписи обычно получается не вполне удовлетворительным и соответствует качеству звучания простейших элек-

тромузыкальных инструментов с окрасом характерным для электронной музыки. В то же время данный метод копирования обеспечивает весьма компактный код, поэтому он нашёл применение ещё в те годы, когда ресурсы средств вычислительной техники были явно недостаточны.

2. *Метод таблично волнового (Wave-Table) синтеза* лучше соответствует современному уровню развития техники. В заранее подготовленных таблицах хранятся образцы звуков для множества различных музыкальных инструментах. В технике такие образцы называют сэмплами. Числовые коды выражают тип инструмента, номер его модели, высоту тона, продолжительность и интенсивность звука, динамику его изменения, некоторые параметры среды, в которой происходит звучание, а также прочие параметры, характеризующие особенности звучания. Поскольку в качестве образцов исполняются реальные звуки, то его качество получается очень высоким и приближается к качеству звучания реальных музыкальных инструментов.

1.8. Носители информации

Информация всегда связана с материальным носителем.

Носитель информации — среда для записи и хранения информации.

Носителем информации может быть:

-любой материальный предмет (бумага, камень, дерево, стол, классная доска, звездная пыль, мусор на полу и т. д.);

-волны различной природы: акустическая (звук), электромагнитная (свет, радиоволна), гравитационная (давление, притяжение) и т. д.;

-вещество в различном состоянии: концентрация молекул в жидком растворе, температура и давление газа и т. д.;

-машинные носители информации: перфоленты, перфокарты, магнитные ленты, магнитные диски, оптические диски и т. д.

Любой сигнал, будь то свет костра, телеграмма, код Морзе, написанный текст, несет какое-либо сведение или сообщение, т. е. информацию. Таким образом, сигнал есть способ передачи информации.

Любой переданный сигнал переносится либо энергией, либо веществом. Иначе и быть не может, ведь наш мир материален. Это либо акустическая волна (звук), либо электромагнитное излучение (свет, радиоволна), либо лист бумаги (написанный текст), либо каменная скрижаль с выбитыми на ней магическими знаками. Материальными носителями наследственности являются гены. Но ни переданная энергия, ни посланное вещество, ни гены сами по себе никакого значения не имеют. Они служат лишь носителями информации.

Само слово «сигнал» имеет общий корень с английским sign, что можно перевести как знак, символ. Написанное слово обозначает некоторое понятие и, таким образом, тоже является сигналом.

Человек всегда стремился доступным ему способом зафиксировать сведения о том, что больше всего его волновало. Наши весьма давние предки оставили нам информацию о себе в виде наскальных росписей в пещерах, где они обитали. Отсюда и произошло идеографическое письмо, в основу которого положены идеограммы: «письменный знак (условное изображение или рисунок), соответствующий не отдельному звуку, а целому слову или морфеме».

Человек всегда изыскивал возможность сделать хранилища информации более компактными, что позволяло делать необходимую информацию транспортабельной, более удобной для хранения, а также ограничивал доступ к ней нежелательных лиц. Поэтому и появились носители информации, пригодные для транспортировки (глиняные таблицы, дощечки, папирус, пергамент, бумага... магнитные носители, лазерные носители и т. д.).

С изменением носителей информации изменились и методы нанесения информации непосредственно на носители, что повлекло за собой необходимость изобретения соответствующих технических средств для работы с информацией: «обжиг глины — глиняные таблицы», «бумага — книгопечатание», «магнитные носители — компьютер». К ним относятся устройства, комплектующие внешнюю память компьютера:

1) накопители на гибких магнитных носителях;

2) накопители на жестких магнитных дисках;

3) накопители на компакт-дисках.

В 90-х гг. XX в. появился дешевый и весьма компактный способ хранения информации с возможностью быстрого доступа к ней. Это *лазерные компактные диски* — CD-ROM диски (англ. compact disk read-only memory — память на компакт-диске, предназначенная только для чтения). Один такой диск может вмещать в себя небольшую домашнюю библиотеку в 300—400 томов. Кроме того, он позволяет хранить на нем не только тексты, но и живую речь, музыку и даже движущееся изображение, причем диаметр диска 8—12 см и толщина 1,8 мм. Несущая информацию тончайшая металлическая пластина защищена 2—3 слоями прозрачной пластмассы, благодаря которой диск можно ронять, царапать. Срок годности такого диска — до 100 лет.

1.8.1 Накопители на гибких магнитных дисках

Гибкий диск (англ. floppy disk), или *дискета*, - устройство для хранения небольших объемов информации, представляющее собой гибкий пластиковый диск в защитной оболочке. Используется для переноса данных с одного компьютера на другой и для распространения программного обеспечения.

Дискета состоит из круглой полимерной подложки, покрытой с обеих сторон магнитным окислом и помещенной в пластиковую упаковку, на внутреннюю поверхность которой нанесено очищающее покрытие. В упаковке сделаны с двух сторон радиальные прорези, через которые головки считывания/записи накопителя получают доступ к диску.

Способ записи двоичной информации на магнитной среде называется магнитным кодированием. Он заключается в том, что магнитные домены в среде выстраиваются вдоль дорожек в направлении приложенного магнитного поля своими северными и южными полюсами. Обычно устанавливается однозначное соответствие между двоичной информацией и ориентацией магнитных доменов.

Информация записывается по концентрическим дорожкам, трекам (темно-зеленый цвет), которые делятся на секторы (светло-зеленый цвет). Количество дорожек и секторов зависит от типа и формата дискеты. Сектор хранит минимальную порцию информации, которая может быть записана на диск или считана. Емкость сектора постоянна и составляет 512 байт. На дискете можно хранить от 360 Кбайт до 2,88 Мбайт информации.

В настоящее время наибольшее распространение получили дискеты со следующими характеристиками:

-диаметр 3,5 дюйма (89 мм);

-емкость 1,44 Мбайт;

-число дорожек 80;

-количество секторов на дорожках 18.

Дискета устанавливается в накопитель на гибких магнитных дисках (floppy-disk drive), автоматически в нем фиксируется, после чего механизм накопителя раскручивается до частоты вращения 360 мин-1. В накопителе вращается сама дискета, магнитные головки остаются неподвижными. Дискета вращается только при обращении к ней.

Накопитель связан с процессором через контроллер гибких дисков.

Контроллер - устройство, которое связывает периферийное оборудование или каналы связи с центральным процессором, освобождая процессор от непосредственного управления функционированием данного оборудования.

1.8.2 Накопители на жестких магнитных дисках

Если гибкие диски - средство переноса данных между компьютерами, то жесткий диск - информационный склад компьютера.

*Накопитель на жестких магнитных диска*х (HDD - Hard Disk Drive), или *винчестер*, - это наиболее массовое запоминающее устройство большой емкости, в котором носителями информации являются круглые алюминиевые пластины - платтеры, обе поверхности которых покрыты слоем магнитного

материала. Используется для постоянного хранения информации - программ и данных.

Как и у дискеты, рабочие поверхности платтеров разделены на кольцевые концентрические дорожки, а дорожки на секторы.

Головки считывания-записи вместе с их несущей конструкцией и дисками заключены в герметический закрытый корпус, называемый модулем данных. При установке модуля данных на дисковод он автоматически соединяется с системой, подкачивающей очищенный охлажденный воздух.

Поверхность платтера имеет магнитное покрытие толщиной всего лишь 1,1 мкм, а также слой смазки для предохранения головки от повреждения при опускании и подъеме на ходу. При вращении платтера над ним образуется воздушный слой, который обеспечивает воздушную подушку для зависания головки на высоте 0,5 мкм над поверхностью диска.

Винчестерские накопители имеют очень большую емкость - от сотен Мегабайт до десятков Гигабайт. У современных моделей скорость вращения шпинделя достигает 7200 об/мин, среднее время поиска данных 10 мс, максимальная скорость передачи данных до 40 Мегабайт/с. В отличие от дискеты, винчестерский диск вращается непрерывно. Винчестерский накопитель связан с процессором через контроллер жесткого диска. Все современные накопители снабжаются встроенным КЭШем (64 Кбайт и более), который существенно повышает их производительность.

1.8.3 Накопители на компакт-дисках

Компакт-диск, или *CD-ROM* (читается "си-ди-ром", сокращение от англ. Compact Disk Read Only Memory - компакт-диск, из которого можно только читать), - это ПЗУ, выполненное с использованием специальной оптической технологии. В ряду запоминающих устройств занимает место где-то между флоппи- и жестким дисками, являясь одновременно и мобильным, и очень емким.

CD-ROM состоит из прозрачной полимерной основы диаметром 12 см и толщиной 1,2 мм. Одна сторона покрыта тонким алюминиевым слоем, защищенным от повреждений слоем лака. Двоичная информация представляется последовательным чередованием углублений (Pits - ямки) и основного слоя (Land - земля).

На одном дюйме (2,54 см) по радиусу диска размещается 16 тысяч дорожек с информацией. Для сравнения: на дюйме по радиусу дискеты всего лишь 96 дорожек. Емкость CD до 780 Мбайт. Информация заносится на диск один раз и не может быть изменена.

Достоинства CD-ROM:

-при малых физических размерах CD-ROM обладают высокой информационной емкостью, что позволяет использовать их в справочных системах и в учебных комплексах с богатым иллюстративным материалом; один CD, имея размеры примерно дискеты, по информационному объему равен почти 500 таким дискетам;

-считывание информации с CD происходит с высокой скоростью, сравнимой со скоростью работы винчестера;

-CD просты и удобны в работе, практически не изнашиваются;

-CD не могут быть поражены вирусами;

-на CD-ROM невозможно случайно стереть информацию;

-стоимость хранения данных (в расчете на 1 Мбайт) низкая.

В отличие от магнитных дисков CD имеют не множество кольцевых дорожек, а одну спиральную, как у грампластинок. Поэтому угловая скорость вращения диска непостоянна: она линейно уменьшается в процессе продвижения читающей магнитной головки к центру диска.

Для работы с CD-ROM нужно подключить к компьютеру накопитель CD-ROM (CD-ROM Drive), в котором CD сменяются как в обычном проигрывателе. Накопители CD-ROM часто называют проигрывателями CD-ROM или приводами CD-ROM.

Участки CD, на которых записаны символы "0" и "1", отличаются коэффициентом отражения лазерного луча, посылаемого накопителем CD-ROM. Эти отличия улавливаются фотоэлементом, и общий сигнал преобразуется в соответствующую последовательность нулей и единиц. Многие накопитель CD-ROM способны воспроизводить обычные аудио-CD. Это позволяет пользователю, работающему за компьютером, слушать музыку в фоновом режиме.

Со временем на смену CD-ROM пришли *цифровые видеодиски DVD* (читается "ди-ви-ди"). Эти диски имеют тот же размер, что и обычные CD, но вмещают 4,7 Гигабайт данных, то есть по объему заменяют 7 стандартных дисков CD-ROM. В скором времени емкость дисков DVD возрастет до 17 Гигабайт. На таких дисках выпускаются полноэкранные видеофильмы отличного качества, программы-тренажеры, мультимедийные игры и многое другое.

Главный недостаток накопителей CD-ROM по сравнению с винчестерскими накопителями - невозможность перезаписи информации.

Тема 2. Принцип работы компьютера – 3 часа

2.1. Что такое компьютер?

Компьютер (англ. *computer* — вычислитель) представляет собой программируемое электронное устройство, способное обрабатывать данные и производить вычисления, а также выполнять другие задачи манипулирования символами

Существует *два основных класса ко*мпьютеров:

- *цифровые компьютеры*, обрабатывающие данные в виде числовых двоичных кодов;
- *аналоговые компьютеры*, обрабатывающие непрерывно меняющиеся физические величины (электрическое напряжение, время и т.д.), которые являются аналогами вычисляемых величин.

Поскольку в настоящее время подавляющее большинство компьютеров являются цифровыми, далее будем рассматривать только этот класс компьютеров и слово "*компьютер*" употреблять в значении "цифровой компьютер". Основу компьютеров образует *аппаратура (HardWare),* построенная, в основном, с использованием электронных и электромеханических элементов и устройств. Принцип действия компьютеров состоит в выполнении *программ (SoftWare)* — заранее заданных, четко определённых последовательностей арифметических, логических и других операций.

Любая компьютерная *программа* представляет собой последовательность отдельных *команд*.

Команда — это описание операции, которую должен выполнить компьютер. Как правило, у команды есть свой код (условное обозначение), исходные данные (операнды) и результат.

Например, у команды "сложить два числа" операндами являются слагаемые, а результатом — их сумма. А у команды "стоп" операндов нет, а результатом является прекращение работы программы.

Результат команды вырабатывается по точно определенным для данной команды правилам, заложенным в конструкцию компьютера.

Совокупность команд, выполняемых данным компьютером, называется *системой команд* этого компьютера.

Компьютеры работают с очень высокой скоростью, составляющей миллионы - сотни миллионов операций в секунду.

2.2. Как устроен компьютер?

Разнообразие современных компьютеров очень велико. Но их структуры основаны на *общих логических при***нципах**, позволяющих выделить в любом компьютере следующие *главные устройства*:

- *память* (запоминающее устройство, ЗУ), состоящую из перенумерованных ячеек;
- *процессор*, включающий в себя *устройство управления* (УУ) и *арифметико-логическое устройство* (АЛУ);

- *устройство ввода*;
- *устройство вывода*.

Эти устройства соединены *каналами связи*, по которым передается информация.

Функции *памяти***:**

- *приём информации* из других устройств;
- *запоминание информации*;
- *выдача информации* по запросу в другие устройства машины.

Функции *процессора***:**

-обработка данных по заданной программе путем выполнения арифметических и логических операций;

-программное управление работой устройств компьютера.

Та часть процессора, которая выполняет команды, называется *арифметико-логическим устройством* (АЛУ), а другая его часть, выполняющая функции управления устройствами, называется *устройством управления* (УУ).

Обычно эти два устройства выделяются чисто условно, *конструктивно они не разделены*.

В составе процессора имеется ряд специализированных дополнительных ячеек памяти, называемых *регистрами***.**

Регистр выполняет функцию кратковременного хранения числа или команды. Над содержимым некоторых регистров специальные электронные схемы могут выполнять некоторые манипуляции. Например, "вырезать" отдельные части команды для последующего их использования или выполнять определенные арифметические операции над числами.

Основным элементом регистра является электронная схема, называемая *триггером***,** которая способна хранить одну двоичную цифру (*разряд*).

Регистр представляет собой совокупность триггеров, связанных друг с другом определённым образом общей системой управления.

Существует несколько типов регистров, отличающихся видом выполняемых операций.

Некоторые важные регистры имеют свои названия, например:

- *сумматор* — регистр АЛУ, участвующий в выполнении каждой операции;
- *счетчик команд* — регистр УУ, содержимое которого соответствует адресу очередной выполняемой команды; служит для автоматической выборки программы из последовательных ячеек памяти;
- *регистр команд* — регистр УУ для хранения кода команды на период времени, необходимый для ее выполнения. Часть его разрядов используется для хранения *кода операции*, остальные — для хранения *кодов адресов операндов*.

Рассмотрим устройство компьютера на примере самой распространенной компьютерной системы — персонального компьютера.

Персональным компьютером (ПК) называют сравнительно недорогой универсальный микрокомпьютер, рассчитанный на одного пользователя.

Персональные компьютеры обычно проектируются на основе принципа открытой архитектуры.

Принцип открытой архитектуры заключается в следующем:

Регламентируются и стандартизируются только *описание принципа действия компьютера и его конфигурация* (определенная совокупность аппаратных средств и соединений между ними). Таким образом, *компьютер можно собирать из отдельных узлов и деталей, разработанных и изготовленных независимыми фирмами-изготовителями*.

Компьютер *легко расширяется и модернизируется* за счёт наличия внутренних расширительных гнёзд, в которые пользователь может вставлять разнообразные устройства, удовлетворяющие заданному стандарту, и тем самым *устанавливать конфигурацию своей машины в соответствии со своими личными предпочтениями.*

Для того чтобы соединить друг с другом различные устройства компьютера, они должны иметь одинаковый *интерфейс* (англ. *interface* от *inter* — между, и *face* — лицо).

Интерфейс — это средство сопряжения двух устройств, в котором все физи-

ческие и логические параметры согласуются между собой.

Если интерфейс является общепринятым, например, утверждённым на уровне международных соглашений, то он называется *стандартным*.

Каждый из функциональных элементов (память, монитор или другое устройство) связан с шиной определённого типа — адресной, управляющей или шиной данных.

Для согласования интерфейсов периферийные устройства подключаются к шине не напрямую, а через свои *контроллеры* (адаптеры) и *порты<.font>* примерно по такой схеме:

Контроллеры и адаптеры представляют собой наборы электронных цепей, которыми снабжаются устройства компьютера с целью совместимости их интерфейсов. Контроллеры, кроме этого, осуществляют непосредственное управление периферийными устройствами по запросам микропроцессора.

Порты устройств представляют собой некие электронные схемы, содержащие один или несколько *регистров ввода-вывода* и позволяющие подключать периферийные устройства компьютера к внешним шинам микропроцессора.

Портами также называют *устройства стандартного интерфейса*: последовательный, параллельный и игровой порты (или интерфейсы).

Последовательный порт обменивается данными с процессором побайтно, а с внешними устройствами — побитно.

Параллельный порт получает и посылает данные побайтно.

К *последовательному* порту обычно подсоединяют медленно действующие или достаточно удалённые устройства, такие, как мышь и модем. К *параллельному* порту подсоединяют более "быстрые" устройства — принтер и сканер. Через *игровой* порт подсоединяется джойстик. Клавиатура и монитор подключаются к своим *специализированным* портам, которые представляют собой просто *разъёмы*.

Основные электронные компоненты, определяющие архитектуру процессора, размещаются на основной плате компьютера, которая называется *системной* или *материнской*. А контроллеры и адаптеры дополнительных

устройств, либо сами эти устройства, выполняются в виде *плат расширения* (*DaughterBoard* — дочерняя плата) и подключаются к шине с помощью *разъёмов расширения*, называемых также *слотами расширения* (англ. *slot* — щель, паз).

2.3. Принципы построения компьютеров

В основу построения подавляющего большинства компьютеров положены следующие общие принципы, сформулированные в 1945 г. американским ученым Джоном фон Нейманом.

1) Принцип программного управления

Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.

Выборка программы из памяти осуществляется с помощью *счетчика команд*. Этот регистр процессора последовательно увеличивает хранимый в нем адрес очередной команды на длину команды. А так как команды программы расположены в памяти друг за другом, то тем самым организуется выборка цепочки команд из последовательно расположенных ячеек памяти.

Если же нужно после выполнения команды перейти не к следующей, а к какой-то другой, используются команды *условного* или *безусловного переходов*, которые заносят в счетчик команд номер ячейки памяти, содержащей следующую команду. Выборка команд из памяти прекращается после достижения и выполнения команды *“стоп”*.

Таким образом, процессор исполняет программу автоматически, без вмешательства человека.

2) Принцип однородности памяти

Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке памяти — число,

текст или команда. Над командами можно выполнять такие же действия, как и над данными.

Это открывает целый ряд возможностей. Например, программа в процессе своего выполнения также может подвергаться переработке**,** что позволяет задавать в самой программе правила получения некоторых ее частей (так в программе организуется выполнение циклов и подпрограмм).

Более того, команды одной программы могут быть получены как результаты исполнения другой программы. На этом принципе основаны методы трансляции — перевода текста программы с языка программирования высокого уровня на язык конкретной машины**.**

3) Принцип адресности

Структурно основная память состоит из перенумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка.

Отсюда следует возможность давать имена областям памяти, так, чтобы к запомненным в них значениям можно было впоследствии обращаться или менять их в процессе выполнения программ с использованием присвоенных имен.

Компьютеры, построенные на этих принципах, относятся к типу *фон-неймановских.*

Но существуют компьютеры, принципиально отличающиеся от фон-неймановских. Для них, например, может *не выполняться принцип программного управления*, т.е. они могут работать без “счетчика команд”, указывающего текущую выполняемую команду программы. Для обращения к какой-либо переменной, хранящейся в памяти, этим компьютерам *не обязательно давать ей имя*. Такие компьютеры называются *не-фон-неймановскими.* Примером такого компьютера является *нейрокомпьютер*.

2.4. Архитектура и структура компьютера

При рассмотрении компьютерных устройств принято различать их архитектуру и структуру.

Архитектурой компьютера называется его описание на некотором общем уровне, включающее описание пользовательских возможностей программирования, системы команд, системы адресации, организации памяти и т.д. Архитектура определяет принципы действия, информационные связи и взаимное соединение основных логических узлов компьютера: процессора, оперативного ЗУ, внешних ЗУ и периферийных устройств. Общность архитектуры разных компьютеров обеспечивает их совместимость с точки зрения пользователя.

Структура компьютера — это совокупность его функциональных элементов и связей между ними. Элементами могут быть самые различные устройства — от основных логических узлов компьютера до простейших схем. Структура компьютера графически представляется в виде структурных схем, с помощью которых можно дать описание компьютера на любом уровне детализации.

Наиболее распространены следующие архитектурные решения.

Классическая архитектура (архитектура фон Неймана) — *одно арифметико-логическое устройство* (АЛУ), через которое проходит *поток данных*, и *одно устройство управления* (УУ), через которое проходит *поток команд — программа*. Это *однопроцессорный компьютер*.

К этому типу архитектуры относится и архитектура персонального компьютера с *общей шиной***.** Все функциональные блоки здесь связаны между собой общей шиной, называемой также *системной магистралью*.

Физически *магистраль* представляет собой многопроводную линию с гнездами для подключения электронных схем. Совокупность проводов магистрали разделяется на отдельные группы: шину адреса, шину данных и шину управления.

Периферийные устройства (принтер и др.) подключаются к аппаратуре компьютера через специальные *контроллеры* — устройства управления периферийными устройствами.

Контроллер — устройство, которое связывает периферийное оборудование или каналы связи с центральным процессором, освобождая процессор от непосредственного управления функционированием данного оборудования.

Многопроцессорная архитектура

Наличие в компьютере *нескольких процессоров* означает, что параллельно может быть организовано много потоков данных и много потоков команд. Таким образом, параллельно могут выполняться несколько фрагментов одной задачи. Структура такой машины, имеющей *общую оперативную память* и несколько процессоров, представлена на рис.

Многомашинная вычислительная система

Здесь *несколько процессоров*, входящих в вычислительную систему, не имеют общей оперативной памяти, а *имеют каждый свою (локальную*). Каждый компьютер в многомашинной системе имеет классическую архитектуру, и такая система применяется достаточно широко.

Однако эффект от применения такой вычислительной системы может быть получен только при решении задач, имеющих очень специальную структуру: она должна разбиваться на столько слабо связанных подзадач, сколько компьютеров в системе.

Преимущество в быстродействии многопроцессорных и многомашинных вычислительных систем перед однопроцессорными очевидно.

Архитектура с параллельными процессорами

Здесь *несколько АЛУ работают под управлением одного УУ*. Это означает, что множество данных может обрабатываться по одной программе— то есть по одному потоку команд.

Высокое быстродействие такой архитектуры можно получить только на задачах, в которых одинаковые вычислительные операции выполняются

одновременно на различных однотипных наборах данных. Структура таких компьютеров представлена на рис. 4.

Рисунок 4 – Архитектура с параллельным процессором

2.5. Основные блоки компьютера

В современных машинах часто присутствуют элементы различных типов архитектурных решений. Существуют и такие архитектурные решения, которые радикально отличаются от рассмотренных выше.

Современный персональный компьютер состоит из нескольких основных конструктивных компонент:

- *системного блока;*
- *монитора;*
- *клавиатуры;*
- *манипуляторов.*

В системном блоке размещаются:

- *блок питания;*
- *накопитель на жёстких магнитных дисках;*
- *накопитель на гибких магнитных дисках;*
- *системная плата;*
- *платы расширения;*
- *накопитель CD-ROM;*
- *и др.*

Корпус системного блока может иметь горизонтальную (*DeskTop*) или вертикальную (*Tower* — башня) компоновку.

Устройство системного блока:

1 — *Системная плата.*

2 — Разъём дополнительного второго процессора.

3 — Центральный процессор с радиатором для отвода тепла.

4 — Разъёмы оперативной памяти.

5 — Накопитель на гибких магнитных дисках.

6 — Накопитель CD-ROM.

7 — Сетевая карта.

8 — Графический акселератор.

9 — *Блок питания*, преобразующий переменное напряжение электросети в постоянное напряжение различной полярности и величины, необходимое для питания системной платы и внутренних устройств. Блок питания содержит *вентилятор*, создающий циркулирующие потоки воздуха для охлаждения системного блока.

Вместо термина ***"системный блок"*** иногда употребляют термин ***"платформа"***.

2.6 Что такое команда?

Команда— это описание элементарной операции, которую должен выполнить компьютер.

В общем случае, команда содержит следующую информацию:

- *код* выполняемой операции;
- указания по определению *операндов* (или их адресов);
- указания по размещению получаемого *результата*.

В зависимости от количества операндов, *команды бывают*:

- одноадресные;
- двухадресные;
- трехадресные;
- переменноадресные.

Команды хранятся в ячейках памяти ***в*** *двоичном коде*.

В современных компьютерах *длина команд переменная* (обычно от двух до четырех байтов), а способы указания адресов переменных весьма разнообразные.

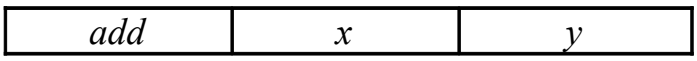
В адресной части команды может быть указан, например:

- сам операнд (число или символ);
- адрес операнда (номер байта, начиная с которого расположен операнд);
- адрес адреса операнда (номер байта, начиная с которого расположен адрес операнда), и др.

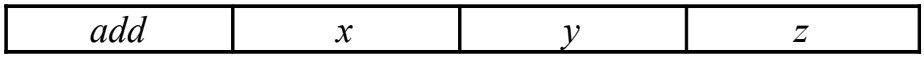
Рассмотрим несколько *возможных вариантов команды сложения* (англ. *add* — сложение), при этом вместо цифровых кодов и адресов будем пользоваться условными обозначениями:

одноадресная команда add x (содержимое ячейки ***x*** сложить с содержимым сумматора, а результат оставить в сумматоре)

двухадресная команда add x, y (сложить содержимое ячеек ***x*** и ***y***, а результат поместить в ячейку ***y***)



трехадресная команда add x, y, z (содержимое ячейки ***x*** сложить с содержимым ячейки ***y***, сумму поместить в ячейку ***z***)



2.7 Как выполняется команда?

Как правило, этот процесс разбивается на следующие этапы:

- из ячейки памяти, адрес которой хранится в счетчике команд, выбирается очередная команда; содержимое счетчика команд при этом увеличивается на длину команды;
- выбранная команда передается в устройство управления на регистр команд;
- устройство управления расшифровывает адресное поле команды;

по сигналам УУ операнды считываются из памяти и записываются в АЛУ на специальные регистры операндов;

- УУ расшифровывает код операции и выдает в АЛУ сигнал выполнить соответствующую операцию над данными;

- результат операции либо остается в процессоре, либо отправляется в память, если в команде был указан адрес результата;
- все предыдущие этапы повторяются до достижения команды “*стоп*”.

2.8 Принцип запоминаемой программы

Для реализации принципа *микропрограммируемости* необходимо наличие в компьютере постоянной памяти, в ячейках которой будут постоянно храниться коды, соответствующие различным комбинациям управляющих сигналов. Каждая такая комбинация позволяет выполнить элементарную операцию, то есть подключить определенные электрические цепи и схемы.

Один набор элементарных операций обеспечит подключение одних электрических цепей и схем и выполнение определенной функции. Изменив в этом же наборе порядок следования элементарных операций, получим другую последовательность подключения электронных схем и выполнение уже другой функции и другого действия компьютера. Расширив этот же набор элементарных операций за счет включения других элементарных операций или увеличения количества уже используемых, получим третий вариант технической реализации и иное действие компьютера, и т.д.

Для того чтобы выполнить элементарную операцию, необходимо задать *управляющий сигнал*. Как же было сказано, он храниться в ячейке постоянной памяти, имеющей совершенно определенный, конкретный адрес. Значит, достаточно задать определенную последовательность адресов, чтобы был сформирован набор управляющих сигналов для выполнения элементарных операций. Задает эту последовательность адресов микропрограмма, также хранящаяся в постоянной памяти.

Команда выполняет все действия в зависимости от того, какой код операции находится в этой команде. При реализации принципа микропрограммируемости каждой команде, а точнее коду операции, соответствует своя микропрограмма. Эта микропрограмма, состоящая из адресов ячеек, где хранят-

ся управляющие сигналы, вызывает их на выполнение и обеспечивает подключение необходимых электрических цепей и схем.

Микропрограммируемость – это микропрограммный принцип управления компьютером, когда каждой программе ставится в соответствие своя микропрограмма, хранящаяся в памяти. Например, операция сложения состоит из элементарных операций: передача информации от одного узла к другому, одноразрядный сдвиг и суммирование одного разряда , чтение первого слагаемого, чтение второго слагаемого и т.д. Техническая реализация этого способа проста благодаря тому, что можно легко видоизменить содержание любой операции и, соответственно, команды за счет изменения микропрограммы.

Тема 3. Аппаратура компьютера. Технические средства реализации информационных процессов. – 2 часа

3.1. Общая структура ЭВМ

Структура компьютера – это совокупность его функциональных элементов и связей между ними. Элементами могут быть самые различные устройства - от основных логических узлов компьютера до простейших схем. Структура компьютера графически представляется в виде структурных схем, с помощью которых можно дать описание компьютера на любом уровне детализации.

Классическая структура (Структура фон Неймана):

Входное устройство служит для ввода в машину всей информации для решения задач. Эта информация состоит из некоторой программы и массива данных, с которыми программа будет работать. В подавляющем большинстве машин и программа, и данные закодированы как числа в бинарной системе счисления. Вся вводимая информация попадает в *запоминающее устройство* или память ЭВМ, где она храниться до момента, когда понадобиться.

Выходное устройство выводит полученные результаты пользователю. Как правило, полученные данные сообщаются пользователю в удобной для него форме (например, в виде текста на языке, близкому к обычному человеческому языку, графиков, рисунков и т. п.).

Ввод информации пользователем в современных компьютерах происходит в удобной для него форме. А *перекодирование* ее в машинное представление реализуется автоматически.

Для того чтобы общение с машиной было удобным, входное и выходное устройства должны содержать специальные средства для перекодирования информации. Поскольку во входном устройстве кодирование осуществляется в машинное представление, а в выходном - из машинного представления, то эти средства могут быть общими. Именно поэтому входное и выходное устройства часто объединяются в единое устройство ввода-вывода. С его помощью реализуется интерфейс (общение) пользователя с машиной.

3.2. Устройство компьютера

При всем многообразии модификаций и вариантов персональных компьютеров в любой, даже самый экзотический комплект неизменно входят одни и те же виды устройств. Делятся эти устройства на *внутренние* (их называют "комплектующими") и *внешние*, периферийные.

Центральная, основная часть любого компьютера, содержащая в себе практически все основные устройства, - *системный блок*, к которому подключены устройства ввода-вывода информации - монитор, клавиатура и мышь.

Все это - обязательный набор, без которого сама работа с компьютером становится невозможной. Правда, первые компьютеры обходились без роскоши типа мыши или монитора: первые модели "персоналок" представляли собой лишь системный блок, снабженный клавиатурой. А роль монитора выполнял обыкновенный телевизор. А присутствие мыши стало обязательным только в эпоху Windows.

Но существует еще масса *дополнительных внешних устройств* - принтеров, сканеров, емких внешних дисководов. Их присутствие не является обязательным для компьютера, но они могут сделать вашу работу более комфортной, подарить вашему ПК новые возможности.

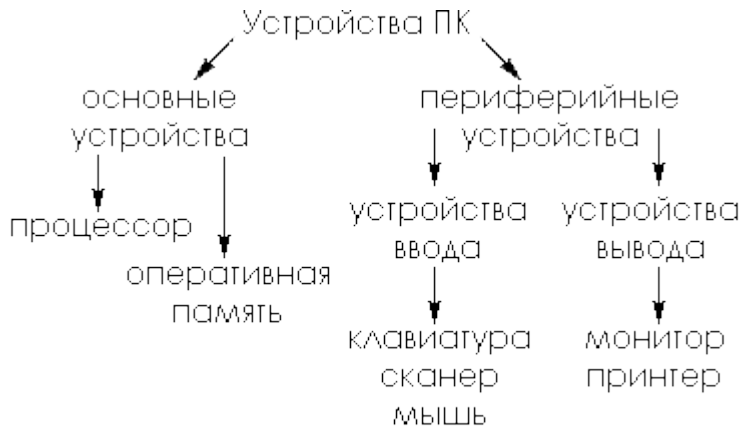


Рисунок 5 – Схема устройства компьютера

3.3. Системный блок

Системный блок представляет собой основной узел, внутри которого установлены наиболее важные компоненты. Устройства, находящиеся внутри системного блока, называют *внутренними*, а устройства, подключаемые к нему снаружи, называют *внешними*. Внешние дополнительные устройства, предназначенные для ввода, вывода и длительного хранения данных, также называют периферийными.

По *внешнему виду* системные блоки различаются формой корпуса. Корпуса персональных компьютеров выпускают в *горизонтальном (desktop)* и *вертикальном (tower)* исполнении. Корпуса, имеющие вертикальное исполнение, различают по габаритам: полноразмерный (big tower), среднеразмерный (midi tower) и малоразмерный (mini tower). Среди корпусов, имеющих горизонтальное исполнение, выделяют плоские и особо плоские (slim). Выбор того или иного типа корпуса определяется вкусом и потребностями модернизации компьютера. Наиболее оптимальным типом корпуса для

большинства пользователей является корпус типа mini tower. Он имеет небольшие габариты, его удобно располагать как на рабочем столе, так и на тумбочке вблизи рабочего стола или на специальном держателе. Он имеет достаточно места для размещения от пяти до семи плат расширения.

Кроме формы, для корпуса важен параметр, называемый форм-фактором. От него зависят требования к размещаемым устройствам. В настоящее время в основном используются корпуса двух форм-факторов: AT и ATX. Форм-фактор корпуса должен быть обязательно согласован с форм-фактором главной (системной) платы компьютера, так называемой материнской платы.

Корпуса персональных компьютеров поставляются вместе с блоком питания и, таким образом, мощность блока питания также является одним из параметров корпуса. Для массовых моделей достаточной является мощность блока питания 200-250 Вт.

3.3.1. Материнская плата

В предыдущем разделе мы выяснили, что первым элементом системного блока является корпус. Вторым и самым важным элементом является *материнская плата*.

Материнская плата (mother board) – основная плата персонального компьютера, представляющая из себя лист стеклотекстолита, покрытый медной фольгой. Путем травления фольги получают тонкие медные проводники соединяющие электронные компоненты. На материнской плате размещаются:

-*процессор* – основная микросхема, выполняющая большинство математических и логических операций;

-*шины* – наборы проводников, по которым происходит обмен сигналами между внутренними устройствами компьютера;

-*оперативная память* (оперативное запоминающее устройство, ОЗУ) – набор микросхем, предназначенных для временного хранения данных, когда компьютер включен;

-*ПЗУ* (постоянное запоминающее устройство) – микросхема, предназначенная для длительного хранения данных, в том числе и когда компьютер выключен;

-*микропроцессорный комплект (чипсет)* – набор микросхем, управляющих работой внутренних устройств компьютера и определяющих основные функциональные возможности материнской платы;

-*разъемы* для подключения дополнительных устройств (слоты).

Необходимо хотя бы кратко ознакомиться с этими устройствами, потому что даже при покупке компьютера без этих знаний не обойтись. Перед Вами откроется тайна странных надписей в рекламных объявлениях о продаже компьютеров. Начнем с размеров и марок самой материнской платы. Менялись микропроцессоры, рождались и умирали системные и локальные шины, а вид и размеры материнской платы практически не менялись с 1984 г. Например, размер оригинальной материнской платы IBM PC/AT под названием Baby-AT был равен 217 на 331 мм, а размеры современной материнской платы P3B-F равны 192 мм на 304 мм.

У материнской платы Baby-AT есть один очень большой недостаток. При установке в слоты расширения печатных плат некоторые из них оказывались прямо над микропроцессором. А так как на микропроцессор чаще всего сверху устанавливают радиатор и вентилятор, установка длинных плат расширения была невозможна. В 1995 году была предложена новая схема размещения элементов на системной плате. Через некоторое время был разработан новый стандарт под названием ATX. Первое отличие материнской платы ATX от Baby AT – расположение слотов расширения. На плате Baby AT они располагаются вдоль длинной стороны платы, а на плате ATX разъемы слотов расширения параллельны короткой стороне, что позволяет в любой слот установить полноразмерную плату расширения. Серьезные изменения коснулись разъемов ввода/вывода. В дополнение к стандартным портам предусмотрены порты будущего: USB(Universal Serial Bus – универсальная последовательная шина) и инфракрасный порт (IrDA). Еще одно отличие –

необходимость дополнительного напряжения 3,3 вольта (для платы Baby AT нужны были напряжения 5 и 12 вольт). В 1999 году на рынке большей частью были представлены материнские платы, допускающие работу с тактовой частотой микропроцессора до 500 Мгц. Некоторые из них имеют отличительный признак – наличие гнезда для подключения микропроцессора с нулевым усилием сопряжения – ZIF (Zero Insertion Force) типа Socket 7. В данное гнездо можно ставить как процессор Intel Pentium, так и процессоры Cyrix 6x86, AMD K5 и K6. Более современные платы имеют разъем Slot 1 или Slot 2 для подключения процессоров Pentium II и Pentium III.

Процессор

Процессор (микропроцессор, центральный процессор, CPU) – основная микросхема компьютера, в которой и производятся все вычисления. Он представляет из себя большую микросхему (например, размеры микропроцессора Pentium примерно 5*5*0,5 см), которую можно легко найти на материнской плате. На процессоре установлен большой медный ребристый радиатор, охлаждаемый вентилятором. Конструктивно процессор состоит из ячеек, в которых данные могут не только храниться, но и изменяться. Внутренние ячейки процессора называют регистрами. Важно также отметить, что данные, попавшие в некоторые регистры, рассматриваются не как данные, а как команды, управляющие обработкой данных в других регистрах. Среди регистров процессора есть и такие, которые в зависимости от своего содержания способны модифицировать исполнение команд. Таким образом, управляя засылкой данных в разные регистры процессора, можно управлять обработкой данных. На этом и основано исполнение программ.

С остальными устройствами компьютера, и в первую очередь с оперативной памятью, процессор связан несколькими группами проводников, называемых *шинами*. Основных шин три:

-шина данных,

-адресная шина,

-командная шина.

Адресная шина. У процессоров Intel Pentium (а именно они наиболее распространены в персональных компьютерах) адресная шина 32-разрядная, то есть состоит из 32 параллельных линий. В зависимости от того, есть напряжение на какой-то из линий или нет, говорят, что на этой линии выставлена единица или ноль. Комбинация из 32 нулей и единиц образует 32-разрядный адрес, указывающий на одну из ячеек оперативной памяти. К ней и подключается процессор для копирования данных из ячейки в один из своих регистров.

Шина данных. По этой шине происходит копирование данных из оперативной памяти в регистры процессора и обратно. В компьютерах, собранных на базе процессоров Intel Pentium, шина данных 64-разрядная, то есть состоит из 64 линий, по которым за один раз на обработку поступают сразу 8 байтов.

Шина команд. Для того чтобы процессор мог обрабатывать данные, ему нужны команды. Он должен знать, что следует сделать с теми байтами, которые хранятся в его регистрах. Эти команды поступают в процессор тоже из оперативной памяти, но не из тех областей, где хранятся массивы данных, а оттуда, где хранятся программы. Команды тоже представлены в виде байтов. Самые простые команды укладываются в один байт, однако есть и такие, для которых нужно два, три и более байтов. В большинстве современных процессоров шина команд 32-разрядная (например, в процессоре Intel Pentium), хотя существуют 64-разрядные процессоры и даже 128-разрядные.

Система команд процессора. В процессе работы процессор обслуживает данные, находящиеся в его регистрах, в поле оперативной памяти, а также данные, находящиеся во внешних портах процессора. Часть данных он интерпретирует непосредственно как данные, часть данных – как адресные данные, а часть – как команды. Совокупность всех возможных команд, которые может выполнить процессор над данными, образует так называемую систему команд процессора. Процессоры, относящиеся к одному семейству, имеют одинаковые или близкие системы команд. Процессоры, относящиеся к

разным семействам, различаются по системе команд и невзаимозаменяемыми.

Совместимость процессоров. Если два процессора имеют одинаковую систему команд, то они полностью совместимы на программном уровне. Это означает, что программа, написанная для одного процессора, может исполняться и другим процессором. Процессоры, имеющие разные системы команд, как правило, несовместимы или ограниченно совместимы на программном уровне.

Группы процессоров, имеющих ограниченную совместимость, рассматривают как семейства процессоров. Так, например, все процессоры Intel Pentium относятся к так называемому семейству x86. Родоначальником этого семейства был 16-разрядный процессор Intel 8086, на базе которого собиралась первая модель компьютера IBM PC. Впоследствии выпускались процессоры Intel 80286, Intel 80386, Intel 80486, Intel Pentium 60,66,75,90,100,133; несколько моделей процессоров Intel Pentium MMX, модели Intel Pentium Pro, Intel Pentium II, Intel Celeron, Intel Xeon, Intel Pentium III, Intel Pentium IV и другие. Все эти модели, и не только они, а также многие модели процессоров компаний AMD и Cyrix относятся к семейству x86 и обладают совместимостью по принципу «сверху вниз».

Принцип совместимости «сверху вниз» – это пример неполной совместимости, когда каждый новый процессор «понимает» все команды своих предшественников, но не наоборот. Это естественно, поскольку двадцать лет назад разработчики процессоров не могли предусмотреть систему команд, нужную для современных программ. Благодаря такой совместимости на современном компьютере можно выполнять любые программы, созданные в последние десятилетия для любого из предшествующих компьютеров, принадлежащего той же аппаратной платформе.

Основные параметры процессоров. Основными параметрами процессоров являются: рабочее напряжение, разрядность, рабочая тактовая частота,

коэффициент внутреннего умножения тактовой частоты и размер кэш-памяти.

Рабочее напряжение процессора обеспечивает материнская плата, поэтому разным маркам процессоров соответствуют разные материнские платы (их надо выбирать совместно). По мере развития процессорной техники происходит постепенное понижение рабочего напряжения. Ранние модели процессоров x86 имели рабочее напряжение 5 В. С переходом к процессорам Intel Pentium оно было понижено до 3,3 В, а в настоящее время оно составляет менее 3 В. Причем ядро процессора питается пониженным напряжением 2,2 В. Понижение рабочего напряжения позволяет уменьшить расстояния между структурными элементами в кристалле процессора до десятитысячных долей миллиметра, не опасаясь электрического пробоя. Пропорционально квадрату напряжения уменьшается и тепловыделение в процессоре, а это позволяет увеличивать его производительность без угрозы перегрева.

Разрядность процессора показывает, сколько бит данных он может принять и обработать в своих регистрах за один раз (за один такт). Первые процессоры x86 были 16-разрядными. Начиная с процессора 80386, они имеют 32-разрядную архитектуру. Современные процессоры семейства Intel Pentium остаются 32-разрядными, хотя и работают с 64-разрядной шиной данных (разрядность процессора определяется не разрядностью шины данных, а разрядностью командной шины).

В основе работы процессора лежит тот же *тактовый принцип*, что и в обычных часах. Исполнение каждой команды занимает определенное количество тактов. В настенных часах такты колебаний задает маятник; в ручных механических часах их задает пружинный маятник; в электронных часах для этого есть колебательный контур, задающий такты строго определенной частоты. В персональном компьютере тактовые импульсы задает одна из микросхем, входящая в микропроцессорный комплект (чипсет), расположенный на материнской плате. Чем выше частота тактов, поступающих на процессор, тем больше команд он может исполнить в единицу времени, тем выше его

производительность. Первые процессоры x86 могли работать с частотой не выше 4,77 МГц, а сегодня рабочие частоты, некоторых процессоров уже превосходят 500 миллионов тактов в секунду (500 МГц).

Тактовые сигналы процессор получает от материнской платы, которая, в отличие от процессора, представляет собой не кристалл кремния, а большой набор проводников и микросхем. По чисто физическим причинам материнская плата не может работать со столь высокими частотами, как процессор. Сегодня ее предел составляет 100-133 МГц. Для получения более высоких частот в процессоре происходит внутреннее умножение частоты на коэффициент 3; 3,5; 4; 4,5; 5 и более.

Обмен данными внутри процессора происходит в несколько раз быстрее, чем обмен с другими устройствами, например с оперативной памятью. Для того чтобы уменьшить количество обращений к оперативной памяти, внутри процессора создают буферную область – так называемую *кэш-память*. Это как бы «сверхоперативная память». Когда процессору нужны данные, он сначала обращается в кэш-память, и только если там нужных данных нет, происходит его обращение в оперативную память. Принимая блок данных из оперативной памяти, процессор заносит его одновременно и в кэш-память. «Удачные» обращения в кэш-память называют попаданиями в кэш. Процент попаданий тем выше, чем больше размер кэш-памяти, поэтому высокопроизводительные процессоры комплектуют повышенным объемом кэш-памяти.

Нередко кэш-память распределяют по нескольким уровням. *Кэш первого уровня* выполняется в том же кристалле, что и сам процессор, и имеет объем порядка десятков Кбайт. *Кэш второго уровня* находится либо в кристалле процессора, либо в том же узле, что и процессор, хотя и исполняется на отдельном кристалле. Кэш-память первого и второго уровня работает на частоте, согласованной с частотой ядра процессора.

Кэш-память третьего уровня выполняют на быстродействующих микросхемах типа SRAM и размещают на материнской плате вблизи процессо-

ра. Ее объемы могут достигать нескольких Мбайт, но работает она на частоте материнской платы.

Шинные интерфейсы материнской платы

Связь между всеми собственными и подключаемыми устройствами материнской платы выполняют ее шины и логические устройства, размещенные в микросхемах микропроцессорного комплекта (чипсета). От архитектуры этих элементов во многом зависит производительность компьютера.

ISA. Историческим достижением компьютеров платформы IBM PC стало внедрение почти двадцать лет назад архитектуры, получившей статус промышленного стандарта ISA (Industry Standard Architecture). Она не только позволила связать все устройства системного блока между собой, но и обеспечила простое подключение новых устройств через стандартные разъемы (слоты). Пропускная способность шины, выполненной по такой архитектуре, составляет до 5,5 Мбайт/с, но, несмотря на низкую пропускную способность, эта шина продолжает использоваться в компьютерах для подключения сравнительно «медленных» внешних устройств, например звуковых карт и модемов.

EISA. Расширением стандарта ISA стал стандарт EISA (Extended ISA), отличающийся увеличенным разъемом и увеличенной производительностью (до 32 Мбайт/с). Как и ISA, в настоящее время данный стандарт считается устаревшим. После 2000 года выпуск материнских плат с разъемами ISA/EISA и устройств, подключаемых к ним, прекращается.

VLB. Название интерфейса переводится как локальная шина стандарта VESA (VESA Local Bus). Понятие «локальной шины» впервые появилось в конце 80-х годов. Оно связано тем, что при внедрении процессоров третьего и четвертого поколений (Intel 80386 и Intel 80486) частоты основной шины (в качестве основной использовалась шина ISA/EISA) стало недостаточно для обмена между процессором и . оперативной памятью. Локальная шина, имеющая повышенную частоту, связала между собой процессор и память в обход основной шины. Впоследствии в эту шину «врезали» интерфейс для под-

ключения видеоадаптера, который тоже требует повышенной пропускной способности, – так появился стандарт VLB, который позволил поднять тактовую частоту локальной шины до 50 МГц и обеспечил пиковую пропускную способность до 130 Мбайт/с.

Основным недостатком интерфейса VLB стало то, что предельная частота локальной шины и, соответственно, ее пропускная способность зависят от числа устройств, подключенных к шине. Так, например, при частоте 50 Мгц к шине может быть подключено только одно устройство (видеокарта). Для сравнения скажем, что при частоте 40 Мгц возможно подключение двух, а при частоте 33 МГц – трех устройств.

PCI. Интерфейс PCI (Peripheral Component Interconnect – стандарт подключения, внешних компонентов) был введен в персональных компьютерах, выполненных на базе процессоров Intel Pentium. По своей сути это тоже интерфейс локальной шины, связывающей процессор с оперативной памятью, в которую врезаны разъемы для р. подключения внешних устройств. Для связи с основной шиной компьютера (ISA/EISA) используются специальные интерфейсные преобразователи – мосты PCI (PCI Bridge). В современных компьютерах функции моста PCI выполняют микросхемы микропроцессорного комплекта (чипсета).

Данный интерфейс поддерживает частоту шины 33 МГц и обеспечивает пропускную способность 132 Мбайт/с. Последние версии интерфейса поддерживают частоту до 66 МГц и обеспечивают производительность 264 Мбайт/с для 32-разрядных данных и 528 Мбайт/с для 64-разрядных данных.

Важным нововведением, реализованным этим стандартом, стала поддержка так называемого режима plug-and-play, впоследствии оформившегося в промышленный стандарт на самоустанавливающиеся устройства. Его суть состоит в том, что после физического подключения внешнего устройства к разъему шины PC7 происходит обмен данными между устройством и материнской платой, в результате которого устройство автоматически получает

номер используемого прерывания, адрес порта подключения и номер канала прямого доступа к памяти.

Конфликты между устройствами за обладание одними и теми же ресурсами (номерами прерываний, адресами портов и каналами прямого доступа к памяти) вызывают массу проблем у пользователей при установке устройств, подключаемых к шине ISA. С появлением интерфейса PCI и с оформлением стандарта plug-and-play появилась возможность выполнять установку новых устройств с помощью автоматических программных средств – эти функции во многом были возложены на операционную систему.

FSB. Шина PCI, появившаяся в компьютерах на базе процессоров Intel Pentium как локальная шина, предназначенная для связи процессора с оперативной памятью, недолго оставалась в этом качестве. Сегодня она используется только как шина для подключения внешних устройств, а для связи процессора и памяти, начиная с процессора Intel Pentium Pro используется специальная шина, получившая название Front Side Bus (FSB). Эта шина работает на очень высокой частоте 100-125 МГц. В настоящее время внедряются материнские платы с частотой шины FSB 133 МГц и ведутся разработки плат с частотой до 200 МГц. Частота шины FSB является одним из основных потребительских параметров – именно он и указывается в спецификации материнской платы. Пропускная способность шины FSB при частоте 100 МГц составляет порядка 800 Мбайт/с.

AGP. Видеоадаптер – устройство, требующее особенно высокой скорости передачи данных. Как при внедрении локальной шины VLB, так и при внедрении локальной шины PCI видеоадаптер всегда был первым устройством, «врезаемым» в новую шину. Сегодня параметры шины PCI уже не соответствуют требованиям видеоадаптеров, поэтому для них разработана отдельная шина, получившая название AGP (Advanced Graphic Port – усовершенствованный графический порт). Частота этой шины соответствует частоте шины PCI (33 МГц или 66 МГц), но она имеет много более высокую про-

пускную способность – до 1066 Мбайт/с (в режиме четырехкратного умножения).

PCMCIA (Personal Computer Memory Card International Association – стандарт международной ассоциации производителей плат памяти для персональных компьютеров). Этот стандарт определяет интерфейс подключения плоских карт памяти небольших размеров и используется в портативных персональных компьютерах.

USB (Universal Serial Bus – универсальная последовательная магистраль). Это одно из последних нововведений в архитектурах материнских плат. Этот стандарт определяет способ взаимодействия компьютера с периферийным оборудованием. Он позволяет подключать до 256 различных устройств, имеющих последовательный интерфейс. Устройства могут включаться цепочками (каждое следующее устройство подключается к предыдущему). Производительность шины USB относительно невелика и составляет до 1,5 Мбит/с, но для таких устройств, как клавиатура, мышь, модем, джойстик и т. п., этого достаточно. Удобство шины состоит в том, что она практически исключает конфликты между различным оборудованием, позволяет подключать и отключать устройства в «горячем режиме» (не выключая компьютер) и позволяет объединять несколько компьютеров в простейшую локальную сеть без применения специального оборудования и программного обеспечения.

Параметры *микропроцессорного комплекта (чипсета)* в наибольшей степени определяют свойства и функции материнской платы. В настоящее время большинство чипсетов материнских плат выпускаются на базе двух микросхем, получивших название *«северный мост»* и *«южный мост».*

«Северный мост» управляет взаимосвязью четырех устройств: процессора, оперативной памяти, порта AGP и шины PCI. Поэтому его также называют четырехпортовым контроллером.

«Южный мост» называют также функциональным контроллером. Он выполняет функции контроллера жестких и гибких дисков, функции моста ISA – PCI, контроллера клавиатуры, мыши, шины USB и т. п.

Оперативная память

Оперативная память (RAM – Random Access Memory) – это массив кристаллических ячеек, способных хранить данные. Существует много различных типов оперативной памяти, но с точки зрения физического принципа действия различают *динамическую память (DRAM)* и *статическую память (SRAM).*

Ячейки *динамической памяти (DRAM)* можно представить в виде микроконденсаторов, способных накапливать заряд на своих обкладках. Это наиболее распространенный и экономически доступный тип памяти. Недостатки этого типа связаны, во-первых, с тем, что как при заряде, так и при разряде конденсаторов неизбежны переходные процессы, то есть запись данных происходит сравнительно медленно. Второй важный недостаток связан с тем, что заряды ячеек имеют свойство рассеиваться в пространстве, причем весьма быстро. Если оперативную память постоянно не «подзаряжать», утрата данных происходит через несколько сотых долей секунды. Для борьбы с этим явлением в компьютере происходит постоянная регенерация (освежение, подзарядка) ячеек оперативной памяти. Регенерация осуществляется несколько десятков раз в секунду и вызывает непроизводительный расход ресурсов вычислительной системы.

Ячейки *статической памяти (SRAM)* можно представить как электронные микроэлементы – триггеры, состоящие из нескольких транзисторов. В триггере хранится не заряд, а состояние (включен/выключен), поэтому этот тип памяти обеспечивает . более высокое быстродействие, хотя технологически он сложнее и, соответственно, дороже.

Микросхемы динамической памяти используют в качестве основной оперативной памяти компьютера. Микросхемы статической памяти исполь-

зуют в качестве вспомогательной памяти (так называемой кэш-памяти), предназначенной для оптимизации работы процессора.

Каждая ячейка памяти имеет свой адрес, который выражается числом. В настоящее время в процессорах Intel Pentium и некоторых других принята 32-разрядная адресация, а это означает, что всего независимых адресов может быть 232. Таким образом, в современных компьютерах возможна непосредственная адресация к полю памяти размером 232 = 4 294 967 296 байт (4,3 Гбайт). Однако это отнюдь не означает, что именно столько оперативной памяти непременно должно быть в компьютере. Предельный размер поля оперативной памяти, установленной в компьютере, определяется микропроцессорным комплектом (чипсетом) материнской платы и обычно составляет несколько сот Мбайт.

Одна адресуемая ячейка содержит восемь двоичных ячеек, в которых можно сохранить 8 бит, то есть один байт данных. Таким образом, адрес любой ячейки памяти можно выразить четырьмя байтами.

Представление о том, сколько оперативной памяти должно быть в типовом компьютере, непрерывно меняется. В середине 80-х годов поле памяти размером 1 Мбайт казалось огромным, в начале 90-х годов достаточным считался объем 4 Мбайт, к середине 90-х годов он увеличился до 8 Мбайт, а затем и до 16 Мбайт. Сегодня типичным считается размер оперативной памяти 32-64 Мбайт, но очень скоро эта величина будет превышена в 2-4 раза даже для моделей массового потребления.

Оперативная память в компьютере размещается на стандартных панельках, называемых *модулями*. *Модули оперативной памяти* вставляют в соответствующие разъемы на материнской плате. Если к разъемам есть удобный доступ, то операцию можно выполнять своими руками. Если удобного доступа нет, может потребоваться неполная разборка узлов системного блока, и в таких случаях операцию поручают специалистам.

Конструктивно модули памяти имеют два исполнения – *однорядные (SIMM-модули)* и *двухрядные (DIMM-модули)*. На компьютерах с процессора-

ми Pentium однорядные модули можно применять только парами (количество разъемов для их установки на материнской плате всегда четное), а DIMM-модули можно устанавливать по одному. Многие модели материнских плат имеют разъемы как того, так и другого типа, но комбинировать на одной плате модули разных типов нельзя.

Модуль памяти

Основными *характеристиками модулей оперативной памяти* являются объем памяти и время доступа. SIMM-модули поставляются *объемами* 4,8,16,32 Мбайт, а DIMM-модули – 16, 32, 64, 128 Мбайт и более. *Время доступа* показывает, сколько времени необходимо для обращения к ячейкам памяти – чем оно меньше, тем лучше. Время доступа измеряется в миллиардных долях секунды (наносекундах, нс). Типичное время доступа к оперативной памяти для SIMM-модулей – 50-70 нс. Для современных DIMM-модулей оно составляет 7-10 нс.

Микросхема ПЗУ и система BIOS

В момент включения компьютера в его оперативной памяти нет ничего – ни данных, ни программ, поскольку оперативная память не может ничего хранить без подзарядки ячеек более сотых долей секунды, но процессору нужны команды, в том числе и в первый момент после включения.

Поэтому сразу после включения на адресной шине процессора выставляется стартовый адрес. Это происходит аппаратно, без участия программ (всегда одинаково). Процессор обращается по выставленному адресу за своей первой командой и далее начинает работать по программам.

Этот исходный адрес не может указывать на оперативную память, в которой пока ничего нет. Он указывает на другой тип памяти – постоянное запоминающее устройство (ПЗУ). *Микросхема ПЗУ* способна длительное время хранить информацию, даже когда компьютер выключен. Программы, находящиеся в ПЗУ, называют «зашитыми» – их записывают туда на этапе изготовления микросхемы.

Комплект программ, находящихся в ПЗУ, образует *базовую систему ввода-вывода (BIOS – Basic Input Output System)*. Основное назначение программ этого пакета состоит в том, чтобы проверить состав и работоспособность компьютерной системы и обеспечить взаимодействие с клавиатурой, монитором, жестким диском и дисководом гибких дисков. Программы, входящие в BIOS, позволяют нам наблюдать на экране диагностические сообщения, сопровождающие запуск компьютера, а также вмешиваться в ход запуска с помощью клавиатуры.

Энергонезависимая память CMOS

Выше мы отметили, что работа таких стандартных устройств, как клавиатура, может обслуживаться программами, входящими в BIOS, но такими средствами нельзя обеспечить работу со всеми возможными устройствами. Так, например, изготовители BIOS абсолютно ничего не знают о параметрах наших жестких и гибких дисков, им не известны ни состав, ни свойства произвольной вычислительной системы. Для того чтобы начать работу с другим оборудованием, программы, входящие в состав BIOS, должны знать, где можно найти нужные параметры. По очевидным причинам их нельзя хранить ни в оперативной памяти, ни в постоянном запоминающем устройстве.

Специально для этого на материнской плате есть микросхема «энергонезависимой памяти», по технологии изготовления называемая CMOS. От оперативной памяти она отличается тем, что ее содержимое не стирается во время выключения компьютера, а от ПЗУ она отличается тем, что данные в нее можно заносить и изменять самостоятельно, в соответствии с тем, какое оборудование входит в состав системы. Эта микросхема постоянно подпитывается от небольшой батарейки, расположенной на материнской плате. Заряда этой батарейки хватает на то, чтобы микросхема не теряла данные, даже если компьютер не будут включать несколько лет.

В микросхеме CMOS хранятся данные о гибких и жестких дисках, о процессоре, о некоторых других устройствах материнской платы. Тот факт, что компьютер четко отслеживает время и календарь (даже и в выключенном

состоянии), тоже связан с тем, что показания системных часов постоянно хранятся (и изменяются) в CMOS.

Таким образом, программы, записанные в BIOS, считывают данные о составе оборудования компьютера из микросхемы CMOS, после чего они могут выполнить обращение к жесткому диску, а в случае необходимости и к гибкому, и передать управление тем программам, которые там записаны.

3.3.2. Жесткий диск

Жесткий диск – основное устройство для долговременного хранения больших объемов данных и программ. На самом деле это не один диск, а группа соосных дисков, имеющих магнитное покрытие и вращающихся с высокой скоростью. Таким образом, этот «диск» имеет не две поверхности, как должно быть у обычного плоского диска, а 2n поверхностей, где n – число отдельных дисков в группе.

Над каждой поверхностью располагается головка, предназначенная для чтения-записи данных. При высоких скоростях вращения дисков (90 об/с) в зазоре между головкой и поверхностью образуется аэродинамическая подушка, и головка парит над магнитной поверхностью на высоте, составляющей несколько тысячных долей миллиметра. При изменении силы тока, протекающего через головку, происходит изменение напряженности динамического магнитного поля в зазоре, что вызывает изменения в стационарном магнитном поле ферромагнитных частиц, образующих покрытие диска. Так осуществляется запись данных на магнитный диск.

Операция считывания происходит в обратном порядке. Намагниченные частицы покрытия, проносящиеся на высокой скорости вблизи головки, наводят в ней ЭДС самоиндукции. Электромагнитные сигналы, возникающие при этом, усиливаются и передаются на обработку.

Управление работой жесткого диска выполняет специальное аппаратно-логическое устройство – контроллер жесткого диска. В прошлом оно представляло собой отдельную дочернюю плату, которую подключали к од-

ному из свободных слотов материнской платы. В настоящее время функции контроллеров дисков выполняют микросхемы, входящие в микропроцессорный комплект (чипсет), хотя некоторые виды высокопроизводительных контроллеров жестких дисков по-прежнему поставляются на отдельной плате.

К *основным параметрам* жестких дисков относятся *емкость и производительность*. *Емкость дисков* зависит от технологии их изготовления. В настоящее время большинство производителей жестких дисков используют изобретенную компанией IBM технологию с использованием гигантского магниторезистивного эффекта (GMR – Giant Magnetic Resistance). Теоретический предел емкости одной пластины, исполненной по этой технологии, составляет порядка 20 Гбайт. В настоящее время достигнут технологический уровень 6,4 Гбайт на пластину, но развитие продолжается.

С другой стороны, *производительность жестких дисков* меньше зависит от технологии их изготовления. Сегодня все жесткие диски имеют очень высокий показатель скорости внутренней передачи данных (до 30-250 Мбайт/с), и потому их производительность в первую очередь зависит от характеристик интерфейса, с помощью которого они связаны с материнской платой. В зависимости от типа интерфейса разброс значений может быть очень большим: от нескольких Мбайт/с до 13-16 Мбайт/с для интерфейсов типа EIDE; до 80 Мбайт/с для интерфейсов типа SCSI-я от 50 Мбайт/с и более для наиболее современных интерфейсов типа IEEE 1394.

Кроме скорости передачи данных с производительностью диска напрямую связан параметр среднего времени доступа. Он определяет интервал времени, необходимый для поиска нужных данных, и зависит от скорости вращения диска. Для дисков, вращающихся с частотой 5400 об/мин, среднее время доступа составляет 9-10 мкс, для дисков с частотой 7200 об/мин – 7-8 мкс. Изделия более высокого уровня обеспечивают среднее время доступа к данным 5-6 мкс. Например, жесткий диск емкостью 18.2 Гб фирмы QUANTUM имеет скорость вращения дисков 7200 об/мин, время поиска –

8,5 мкс, скорость внутренней передачи данных – 235 Мбайт/с, размер буфера – 512 Кбайт.

Жесткий диск устанавливается в специальные монтажные отсеки внутри системного блока. Жесткий диск подключается прямо к материнской плате плоским 40 контактным кабелем. К одному кабелю можно подключить два жестких диска или один жесткий диск и один накопитель для чтения компакт-дисков (CD-ROM drive). Ранее для подключения жестких дисков применялись специальные платы расширения – мультикарты. На мультикартах располагались так же разъемы для подключения гибких дисков, разъемы (порты) COM1 и COM2 для подключения мыши, модема, сканера и LPT (от одного до трех) для подключения принтера. Сейчас все эти разъемы располагаются прямо на материнской плате.

3.3.3. Дисковод гибких дисков

Информация на жестком диске может храниться годами, однако иногда требуется ее перенос с одного компьютера на другой. Несмотря на свое название, жесткий диск является весьма хрупким прибором, чувствительным к перегрузкам, ударам и толчкам. Теоретически, переносить информацию с одного рабочего места на другое путем переноса жесткого диска возможно, и в некоторых случаях так и поступают, но все-таки этот прием считается нетехнологичным, поскольку требует особой аккуратности и определенной квалификации.

Для оперативного переноса небольших объемов информации используют так называемые *гибкие магнитные диски (дискеты)*, которые вставляют в специальный накопитель – *дисковод*. Приемное отверстие накопителя находится на лицевой панели системного блока. Правильное направление подачи гибкого диска отмечено стрелкой на его пластиковом кожухе.

Основными параметрами гибких дисков являются: *технологический размер* (измеряется в дюймах), *плотность записи* (измеряется в кратных единицах) и *полная емкость*.

Первый компьютер IBM PC (родоначальник платформы) был выпущен в 1981 году К нему можно было подключить внешний накопитель, использующий односторонние *гибкие диски диаметром 5,25 дюйма*. Емкость диска составляла 160 Кбайт. В следующем году появились аналогичные двусторонние диски емкостью 320 Кбайт. Начиная с 1984 года выпускались гибкие диски 5,25 дюйма высокой плотности (1,2 Мбайт). В наши дни диски размером 5,25 дюйма не используются, и соответствующие дисководы в базовой конфигурации персональных компьютеров после 1994 года не поставляются.

Гибкие диски размером 3,5 дюйма выпускают с 1980 года. Односторонний диск обычной плотности имел емкость 180 Кбайт, двусторонний – 360 Кбайт, а двусторонний двойной плотности – 720 Кбайт. Ныне стандартными считают диски размером 3,5 дюйма высокой плотности. Они имеют емкость 1440 Кбайт (1,4 Мбайт) и маркируются буквами HD (high density – высокая плотность).

С нижней стороны гибкий диск имеет центральную втулку, которая захватывается шпинделем дисковода и приводится во вращение. Магнитная поверхность прикрыта сдвигающейся шторкой для защиты от влаги, грязи и пыли. Если на гибком диске записаны ценные данные, его можно защитить от стирания и перезаписи, сдвинув защитную задвижку так, чтобы образовалось открытое отверстие. Для разрешения записи задвижку перемещают в обратную сторону и перекрывают отверстие. В некоторых случаях для безусловной защиты информации на диске задвижку выламывают физически, но и в этом случае разрешить запись на диск можно, если, например, заклеить образовавшееся отверстие тонкой полоской липкой ленты.

Гибкие диски считаются малонадежными носителями информации. Пыль, грязь, влага, температурные перепады и внешние электромагнитные поля очень часто становятся причиной частичной или полной утраты данных, хранившихся на гибком диске. Поэтому использовать гибкие диски в качестве основного средства хранения информации недопустимо. Их используют

только для транспортировки информации или в качестве дополнительного (резервного) средства хранения.

3.3.4. Дисковод компакт-дисков CD-ROM

В период 1994-1995 годах в базовую конфигурацию персональных компьютеров перестали включать дисководы гибких дисков диаметром 5,25 дюйма, но вместо них стандартной стала считаться установка дисковода CD-ROM, имеющего такие же внешние размеры.

Аббревиатура *CD-ROM* (Compact Disc Read-Only Memory) переводится на русский язык как постоянное запоминающее устройство на основе компакт-диска. Принцип действия этого устройства состоит в считывании числовых данных с помощью лазерного луча, отражающегося от поверхности диска. Цифровая запись на компакт-диске отличается от записи на магнитных дисках очень высокой плотностью, и стандартный компакт-диск может хранить примерно 650 Мбайт данных.

Большие объемы данных характерны для мультимедийной информации (графика, музыка, видео), поэтому дисководы CD-ROM относят к аппаратным средствам мультимедиа. Программные продукты, распространяемые на лазерных дисках, называют мультимедийными изданиями. Сегодня мультимедийные издания завоевывают все более прочное место среди других традиционных видов изданий. Так, например, существуют книги, альбомы, энциклопедии и даже периодические издания (электронные журналы), выпускаемые на CD-ROM.

Основным недостатком стандартных дисководов CD-ROM является невозможность записи данных, но параллельно с ними существуют и устройства однократной записи CD-R (Compact Disk Recorder), и устройства многократной записи CD-RW.

Основным параметром дисководов CD-ROM является *скорость чтения данных*. Она измеряется в кратных долях. За единицу измерения принята скорость чтения в первых серийных образцах, составлявшая 150 Кбайт/с. Та-

ким образом, дисковод с удвоенной скоростью чтения обеспечивает производительность 300 Кбайт/с, с учетверенной скоростью – 600 Кбайт/с и т. д. В настоящее время наибольшее распространение имеют устройства чтения CD-ROM с производительностью 32х-50х. Современные образцы устройств однократной записи имеют производительность 4х-8х, а устройств многократной записи – до 4х.

3.4. Монитор

Монитор – устройство визуального представления данных. Это не единственно возможное, но главное устройство вывода. Его основными *потребительскими параметрами* являются: *размер* и *шаг маски экрана*, *максимальная частота регенерации изображения, класс защиты*.

По способу формирования изображения мониторы делятся на *жидкокристаллические (LCD)* и построенные *на основе электронно-лучевой трубки (CRT).*

Мониторы на электронно-лучевой трубке (CRT)

Изображение на экране монитора получается в результате облучения люминофорного покрытия остронаправленным пучком электронов, разогнанных в вакуумной колбе. Для получения цветного изображения люминофорное покрытие имеет точки или полоски трех типов, светящиеся красным, зеленым и синим цветом. Чтобы на экране все три луча сходились строго в одну точку и изображение было четким, перед люминофором ставят маску – панель с регулярно расположенными отверстиями или щелями. Часть мониторов оснащена маской из вертикальных проволочек, что усиливает яркость и насыщенность изображения. Чем меньше шаг между отверстиями или щелями (шаг маски), тем четче и точнее полученное изображение. *Шаг маски* измеряют в долях миллиметра. В настоящее время наиболее распространены мониторы с шагом маски 0,25-0,27 мм. Устаревшие мониторы могут иметь шаг до 0,43 мм, что негативно сказывается на органах зрения при работе с

компьютером. Модели повышенной стоимости могут иметь значение менее 0,25 мм.

Одним из главных параметров монитора является *частота кадровой развертки*, называемой также *частотой регенерации* (обновления) изображения (частота смены изображения на экране). Она показывает, сколько раз в течение секунды монитор может полностью сменить изображение (поэтому ее также называют частотой кадров). Частоту регенерации изображения измеряют в герцах (Гц). Чем она выше, тем четче и устойчивее изображение, тем меньше утомление глаз, тем больше времени можно работать с компьютером непрерывно. Этот параметр зависит не только от монитора, но и от свойств и настроек видеоадаптера, хотя предельные возможности определяет все-таки монитор. При частоте регенерации порядка 60 Гц мелкое мерцание изображения заметно невооруженным глазом. Сегодня такое значение считается недопустимым. Минимальным считают значение 75 Гц, нормативным – 85 Гц и комфортным – 100 Гц и более. При работе с компьютером нужно помнить, что главная нагрузка приходится на зрение и если изображение будет дрожать на экране глаза будут сильно утомляться.

Размер монитора измеряется между противоположными углами трубки кинескопа по диагонали. Единица измерения – дюймы. Стандартные размеры: 14"; 15"; 17"; 19"; 20"; 21". В настоящее время наиболее универсальными являются мониторы размером 15 и 17 дюймов, а для операций с графикой желательны мониторы размером 19-21 дюйм.

Класс защиты монитора определяется стандартом, которому соответствует монитор с точки зрения требований техники безопасности. В настоящее время общепризнанными считаются следующие международные стандарты: MPR-II, TCO-92, TCO-95, TCO-99 (приведены в хронологическом порядке). Стандарт MPR-II ограничил уровни электромагнитного излучения пределами, безопасными для человека. В стандарте TCO-92 эти нормы были сохранены, а в стандартах TCO-95 и TCO-99 ужесточены. Эргономические и экологические нормы впервые появились в стандарте TCO-95, а стандарт

TCO-99 установил самые жесткие нормы по параметрам, определяющим качество изображения (яркость, контрастность, мерцание, антибликовые свойства покрытия).

Большинством параметров изображения, полученного на экране монитора, можно управлять программно. Программные средства, предназначенные для этой цели, обычно входят в системный комплект программного обеспечения.

Мониторы на жидких кристаллах (LCD)

Мониторы на катодно-лучевой трубке (CRT) устареют в течение ближайших нескольких лет. Их место займут тонкие и плоские дисплеи на жидких кристаллах, до сих пор использовавшиеся в ноутбуках и компьютерах laptop.

Основными достоинствами LCD-мониторов являются:

-более живые и естественные цвета и образы;

-отсутствие искривления экрана;

-меньшее тепловое излучение;

-потребление почти на 65% меньше энергии, чем CRT-мониторы;

-отсутствие электромагнитного излучения (полностью безопасны для здоровья);

-вес примерно в два раза меньше традиционных мониторов

-занимаемая площадь на столе почти в два раза меньше, чем у CRT-мониторов. LCD-дисплеи настолько тонки, что их можно крепить к стене.

Сейчас используется два типа LCD-технологии для создания изображения на экране: *активноматричная*, также называемая технологией на тонкопленочных транзисторах (TFT), и *пассивноматричная*, или матрица с двойным сканированием (DSTN). Более популярными являются активноматричные дисплеи.

Единственный показатель, по которому LCD-мониторы проигрывают CRT-мониторам – это цена. Жидкокристаллический дисплей 15" продается

по цене около 400 - 500 долларов. Тогда как обычный 15" монитор стоит от 150 до 200 долларов. Однако цены падают. Постоянно увеличивающийся спрос должен постепенно снизить цены.

3.5. Клавиатура

Клавиатура – клавишное устройство управления персональным компьютером. Служит для ввода алфавитно-цифровых (знаковых) данных, а также команд управления. Комбинация монитора и клавиатуры обеспечивает простейший интерфейс пользователя. С помощью клавиатуры управляют компьютерной системой, а с помощью монитора получают от нее отклик.

Известно *два типа* клавиатур – клавиатура с *механическими* и *мембранными* переключателями. Переключатель первого типа это обычный механический датчик, традиционное устройство известное уже несколько десятилетий. Второй тип датчика устроен несколько сложнее. Переключатель данного типа представляет из себя набор мембран, при нажатии на клавишу верхняя мембрана прогибается и через специальное отверстие в изолирующей мембране замыкается на нижнюю мембрану. Как правило, предпочтение отдается клавиатуре с механическими датчиками. Они выдерживают многолетнюю эксплуатацию, надежны и поддаются ремонту в случае необходимости.

Принцип действия. Клавиатура относится к стандартным средствам персонального компьютера. Ее основные функции не нуждаются в поддержке специальными системными программами (драйверами). Необходимое программное обеспечение для начала работы с компьютером уже имеется в микросхеме ПЗУ в составе базовой системы ввода-вывода (BIOS), и потому компьютер реагирует на нажатия клавиш сразу после включения.

Физически клавиатура и процессор связаны только двумя проводами, контролер процессора сканирует переключатели клавиш и при нажатии на любую клавишу по этим двум проводам передается уникальный скан-код размером один байт. Когда скан-код попадает в процессор, инициализирует-

ся аппаратное прерывание IRQ 9(Interrupt 9, Int9). Скан-код анализируется процессором и преобразуется в код символа. Далее полученный код символа помещается в небольшую область памяти, известную как буфер клавиатуры. Введенный символ хранится в буфере клавиатуры до тех пор, пока его не заберет оттуда та программа, для которой он и предназначался, например текстовый редактор или текстовый процессор. Если символы поступают в буфер чаще, чем забираются оттуда, наступает эффект переполнения буфера. В этом случае ввод новых символов на некоторое время прекращается. На практике в этот момент при нажатии на клавишу мы слышим предупреждающий звуковой сигнал и не наблюдаем ввода данных.

Из данного объяснения ясно, что каждой клавише присвоен уникальный цифровой код и существуют специальные таблицы кодировки клавиатуры. Например, кодовая таблица США имеет номер 437 (как правило, она записана в специальную микросхему – знакогенератор процессора), а кодовая страница России имеет номер 866. Для смены кодировки клавиатуры применяются специальные программы – клавиатурные драйверы. Современные клавиатуры способны не только передавать данные в процессор, но и воспринимать команды от него.

Состав клавиатуры. Стандартная клавиатура имеет более 100 клавиш, функционально распределенных по нескольким группам:

1) *Группа алфавитно-цифровых клавиш* предназначена для ввода знаковой информации и команд, набираемых по буквам. Каждая клавиша может работать в нескольких режимах (регистрах) и, соответственно, может использоваться для ввода нескольких символов. Переключение между нижним регистром (для ввода строчных символов) и верхним регистром (для ввода прописных символов) выполняют удержанием клавиши SHIFT (нефиксированное переключение). При необходимости жестко переключить регистр используют клавишу CAPS LOCK (фиксированное переключение). Если клавиатура используется для ввода данных, абзац закрывают нажатием клавиши ENTER. При этом автоматически начинается ввод текста с новой строки.

Если клавиатуру используют для ввода команд, клавишей ENTER завершают ввод команды и начинают ее исполнение.

Для разных языков существуют различные схемы закрепления символов национальных алфавитов за конкретными алфавитно-цифровыми клавишами. Такие схемы называются *раскладками клавиатуры*. Переключения между различными раскладками выполняются программным образом – это одна из функций операционной системы. Соответственно, способ переключения зависит от того, в какой операционной системе работает компьютер. Например, в системе Windows 98 для этой цели могут использоваться следующие комбинации: левая клавиша ALT+SHIFT или CTRL+SHIFT. При работе с другой операционной системой способ переключения можно установить по справочной системе той программы, которая выполняет переключение.

Общепринятые раскладки клавиатуры имеют свои корни в раскладках клавиатур пишущих машинок. Для персональных компьютеров IBM PC типовыми считаются раскладки QWERTY (английская) и ЙЦУКЕНГ (русская). Раскладки принято именовать по символам, закрепленным за первыми клавишами верхней строки алфавитной группы.

2) *Группа функциональных клавиш* включает двенадцать клавиш (от F1 до F12), размещенных в верхней части клавиатуры. Функции, закрепленные за данными клавишами, зависят от свойств конкретной работающей в данный момент программы, а в некоторых случаях и от свойств операционной системы. Общепринятым для большинства программ является соглашение о том, что клавиша F1 вызывает справочную систему, в которой можно найти справку о действии прочих клавиш.

3) *Служебные клавиши* располагаются рядом с клавишами алфавитно-цифровой группы. В связи с тем, что ими приходится пользоваться особенно часто, они имеют увеличенный размер. К ним относятся рассмотренные выше клавиши SHIFT и ENTER, регистровые клавиши ALT и CTRL (их используют в комбинации с другими клавишами для формирования команд), клавиша TAB (для ввода позиций табуляции при наборе текста), клавиша

ESC (от английского слова Escape) для отказа от исполнения последней введенной команды и клавиша BACKSPACE для удаления только что введенных знаков (она находится над клавишей ENTER и часто маркируется стрелкой, направленной влево).

Служебные клавиши PRINT SCREEN, SCROLL LOCK и PAUSE/BREAK размещаются справа от группы функциональных клавиш и выполняют специфические функции, зависящие от действующей операционной системы. Общепринятыми являются следующие действия:

PRINT SCREEN – печать текущего состояния экрана на принтере (для MS-DOS) или сохранение его в специальной области оперативной памяти, называемой буфером обмена (для Windows).

SCROLL LOCK – переключение режима работы в некоторых (как правило, устаревших) программах.

PAUSE/BREAK – приостановка/прерывание текущего процесса.

4) Две группы *клавиш управления курсором* расположены справа от алфавитно-цифровой панели.

Курсором называется экранный элемент, указывающий место ввода знаковой информации. Курсор используется при работе с программами, выполняющими ввод данных и команд с клавиатуры. Клавиши управления курсором позволяют управлять позицией ввода.

Четыре клавиши со стрелками выполняют смещение курсора в направлении, указанном стрелкой. Действие прочих клавиш описано ниже.

PAGE UP/PAGE DOWN – перевод курсора на одну страницу вверх или вниз. Понятие «страница» обычно относится к фрагменту документа, видимому на экране. В графических операционных системах (например Windows) этими клавишами выполняют «прокрутку» содержимого в текущем окне. Действие этих клавиш во многих программах может быть модифицировано с помощью служебных регистровых клавиш, в первую очередь SHIFT и CTRL Конкретный результат модификации зависит от конкретной программы и/или операционной системы.

Клавиши HOME и END переводят курсор в начало или конец текущей строки, соответственно. Их действие также модифицируется регистровыми клавишами.

Традиционное назначение клавиши INSERT состоит в переключении режима ввода данных (переключение между режимами вставки и замены). Если текстовый курсор находится внутри существующего текста, то в режиме вставки происходит ввод новых знаков без замены существующих символов (текст как бы раздвигается). В режиме замены новые знаки заменяют текст, имевшийся ранее в позиции ввода.

В современных программах действие клавиши INSERT может быть иным. Конкретную информацию следует получить в справочной системе программы. Возможно, что действие этой клавиши является настраиваемым, – это также зависит от свойств конкретной программы.

Клавиша DELETE предназначена для удаления знаков, находящихся справа от текущего положения курсора. При этом положение позиции ввода остается неизменным.

5) *Группа клавиш дополнительной панели* дублирует действие цифровых и некоторых знаковых клавиш основной панели. Во многих случаях для использования этой группы клавиш следует предварительно включать клавишу-переключатель NUM LOCK (о состоянии переключателей NUM LOCK, CAPS LOCK и SCROLL LOCK можно судить по светодиодным индикаторам, обычно расположенным в правом верхнем углу клавиатуры).

Появление дополнительной панели клавиатуры относится к началу 80-х годов. В то время клавиатуры были относительно дорогостоящими устройствами. Первоначальное назначение дополнительной панели состояло в снижении износа основной панели при проведении расчетно-кассовых вычислений, а также при управлении компьютерными играми (при выключенном переключателе NUM LOCK клавиши дополнительной панели могут использоваться в качестве клавиш управления курсором).

Настройка клавиатуры. Клавиатуры персональных компьютеров обладают свойством повтора знаков, которое используется для автоматизации процесса ввода. Оно состоит в том, что при длительном удержании клавиши начинается автоматический ввод связанного с ней кода. При этом настраиваемыми параметрами являются:

-*интервал времени* после нажатия, по истечении которого начнется автоматический повтор кода;

-*темп повтора* (количество знаков в секунду).

Средства настройки клавиатуры относятся к системным и обычно входят в состав операционной системы. Кроме параметров режима повтора настройке подлежат также используемые раскладки и органы управления, используемые для переключения раскладок.

Со средствами настройки клавиатуры мы познакомимся при изучении функций операционной системы.

Специальные клавиатуры. Клавиатура является основным устройством ввода данных. Специальные клавиатуры предназначены для повышения эффективности процесса ввода данных. Это достигается путем изменения формы клавиатуры, раскладки ее клавиш или метода подключения к системному блоку.

Клавиатуры, имеющие специальную форму, рассчитанную с учетом требований эргономики, называют эргономичными клавиатурами. Их целесообразно применять на рабочих местах, предназначенных для ввода большого количества знаковой информации. Эргономичные клавиатуры не только повышают производительность наборщика и снижают общее утомление в течение рабочего дня, но и снижают вероятность и степень развития ряда заболеваний, например туннельного синдрома кистей рук и остеохондроза верхних отделов позвоночника.

Раскладка клавиш стандартных клавиатур далека от оптимальной. Она сохранилась со времен ранних образцов механических пишущих машин. В настоящее время существует техническая возможность изготовления клавиа-

тур с оптимизированной раскладкой, и существуют образцы таких устройств (в частности, к ним относится клавиатура Дворака). Однако практическое внедрение клавиатур с нестандартной раскладкой находится под вопросом в связи с тем, что работе с ними надо учиться специально. На практике подобными клавиатурами оснащают только специализированные рабочие места.

По методу подключения к системному блоку различают *проводные* и *беспроводные* клавиатуры. Передача информации в беспроводных системах осуществляется инфракрасным лучом. Обычный радиус действия таких клавиатур составляет несколько метров. Источником сигнала является клавиатура.

3.5. Мышь

Мышь – устройство управления манипуляторного типа. Представляет собой плоскую коробочку с двумя-тремя кнопками. Перемещение мыши по плоской поверхности синхронизировано с перемещением графического объекта (указателя мыши) на экране монитора.

Принцип действия. В отличие от рассмотренной ранее клавиатуры, мышь не является стандартным органом управления, и персональный компьютер не имеет для нее выделенного порта. Для мыши нет и постоянного выделенного прерывания, а базовые средства ввода и вывода (BIOS) компьютера, размещенные в постоянном запоминающем устройстве (ПЗУ), не содержат программных средств для обработки прерываний мыши.

В связи с этим в первый момент после включения компьютера мышь не работает. Она нуждается в поддержке специальной системной программы – драйвера мыши. Драйвер устанавливается либо при первом подключении мыши, либо при установке операционной системы компьютера. Хотя мышь и не имеет выделенного порта на материнской плате, для работы с ней используют один из стандартных портов, средства для работы с которыми имеются в составе BIOS. Драйвер мыши предназначен для интерпретации сигналов, поступающих через порт. Кроме того, он обеспечивает механизм передачи

информации о положении и состоянии мыши операционной системе и работающим программам.

Компьютером управляют перемещением мыши по плоскости и кратковременными нажатиями правой и левой кнопок (Эти нажатия называются щелчками.) В отличие от клавиатуры мышь не может напрямую использоваться для ввода знаковой информации – ее принцип управления является событийным. Перемещения мыши и щелчки ее кнопок являются событиями с точки зрения ее программы-драйвера. Анализируя эти события, драйвер устанавливает, когда произошло событие, и в каком месте экрана в этот момент находился указатель. Эти данные передаются в прикладную программу, с которой работает пользователь в данный момент. По ним программа может определить команду, которую имел в виду пользователь, и приступить к ее исполнению.

Комбинация монитора и мыши обеспечивает наиболее современный тип интерфейса пользователя, который называется графическим. Пользователь наблюдает на экране графические объекты и элементы управления. С помощью мыши он изменяет свойства объектов и приводит в действие элементы управления компьютерной системой, а с помощью монитора получает от нее отклик в графическом виде.

Стандартная мышь имеет только две кнопки, хотя существуют нестандартные мыши с тремя кнопками или с двумя кнопками и одним вращающимся регулятором. В последнее время все большее распространение получают мыши с колесиком прокрутки, расположенным между двумя кнопками и позволяющим выполнять прокрутку в любых приложениях Windows. Функции нестандартных органов управления определяются тем программным обеспечением, которое поставляется вместе с устройством.

К числу *регулируемых параметров* мыши относятся: *чувствительность* (выражает величину перемещения указателя на экране при заданном линейном перемещении мыши), *функции левой и правой кнопок*, а также *чувствительность к двойному нажатию* (максимальный интервал времени, при

котором два щелчка кнопкой мыши расцениваются как один двойной щелчок). Программные средства, предназначенные для этих регулировок, обычно входят в системный комплект программного обеспечения – мы рассмотрим их при изучении операционной системы.

*Специальные манип*уляторы. Кроме обычной мыши существуют и другие типы манипуляторов, например: трекболы, пенмаусы, инфракрасные мыши.

Трекбол в отличие от мыши устанавливается стационарно, и его шарик приводится в движение ладонью руки. Преимущество трекбола состоит в том, что он не нуждается в гладкой рабочей поверхности, поэтому трекболы нашли широкое применение в портативных персональных компьютерах.

Пенмаус представляет собой аналог шариковой авторучки, на конце которой вместо пишущего узла установлен узел, регистрирующий величину перемещения.

Инфракрасная мышь отличается от обычной наличием устройства беспроводной связи с системным блоком.

Для компьютерных игр и в некоторых специализированных имитаторах применяют также манипуляторы рычажно-нажимного типа (*джойстики*) и аналогичные им *джой-пады*, *геймпады* и *штурвально-педальные* устройства. Устройства этого типа подключаются к специальному порту, имеющемуся на звуковой карте, или к порту USB.

3.6. Принтер

Принтер – устройство для получения бумажных копий документов. Принтеры бывают *матричные, лазерные, струйные*.

Матричные принтеры позволяют получить самые дешевые копии документов на недорогой бумаге, но качество печати невысоко. Кроме того, матричные принтеры шумят при работе, и печать страницы производится довольно долго.

Для использования в офисе больше всего подходят *лазерные принтеры*, которые позволяют получать высококачественные черно-белые копии документов. Метод нанесения - электростатическое сухое порошковое нанесение изображения. Для печати используется обычная бумага для копировальных аппаратов.

Современный лазерный принтер для офиса Minolta PagePro25, печатает 25 копий в минуту, на бумаге форматов А3, А4 и А5.

Струйные принтеры в настоящее время стали основными устройствами массовой цветной печати. Большинство струйных принтеров используют для печати 4 краски, но уже есть принтеры, использующие 7 красок и обеспечивающие более высококачественную печать. Скорость печати примерно 500 с на страницу.

Подавляющее большинство принтеров, независимо от принципа печати, позволяют выводить на бумагу произвольное изображение. Способы достижения этого разные: в матричных и струйных принтерах можно непосредственно управлять печатающей головкой и вращением вала, в лазерных принтерах можно формировать растровое изображение целиком для всей страницы. Существует лишь одна, довольно редкая теперь, категория принтеров, предназначенных только для печати текста - АЦПУ (алфавитно-цифровые печатающие устройства). В АЦПУ символы отлиты на барабане или выбиты на металлической ленте, поэтому АЦПУ не позволяют менять гарнитуру, размер и другие параметры шрифта, хотя можно заменить барабан или ленту.

Таким образом, на АЦПУ можно печатать только стандартизованные символы, используя их коды, остальные типы принтеров поддерживают как печать в кодах, так и формирование произвольного изображения - графическую печать. Поэтому любой принтер, кроме АЦПУ, может быть русифицирован, так как буквы - это частный случай графических изображений. Другой вопрос, насколько сложной окажется такая русификации.

Русификация принтеров

Русификация принтера зависит от способа печати: кодами символов или в графическом режиме. Вообще, понятие "русификация" имеет смысл только для печати кодами. Разберем этот вопрос подробнее.

Печать кодами является экономичной - в идеале для печати одного символа нужно послать на принтер один байт (в Unicode - два, но мне еще не встречались принтеры, рассчитанные на Unicode), и быстрой. Идея печати кодами очень проста и не отличается от принципа вывода информации на экран. Каждому символу соответствует код, совокупность символов и их кодов образует кодовую таблицу. Для вывода на экран используется текущая кодовая таблица компьютера, для вывода на принтер - кодовая таблица принтера. Ясно, что для того, чтобы распечатанный текст совпадал с текстом на экране, кодовые таблицы компьютера и принтера должны совпадать. Именно это и является основной задачей русификации принтера - обеспечить поддержку кодовой таблицы, используемой в компьютере.

Для каждой страны и каждого языка существует своя кодовая таблица, для некоторых языков даже несколько таблиц, в которых одни и те символы кодируются по-разному, а одним и тем же кодам соответствуют разные буквы. Ситуация усугубляется отличием кодировок в различных операционных системах. В русской версии DOS используется кодовая таблица (code page, CP) с номером 866 , а в русифицированной системе Windows - таблица 1251. В системе Unix и в компьютерах Macintosh - свои кодировки. Лучшее, что можно обещать, это что коды латинских букв будут одни и те же.

Русификация принтера означает, что принтер имеет "прошитые" русские буквы, которые программа не должна "рисовать" сама, и что встроенная в принтер кодовая таблица совпадает с таблицей компьютера. Таким образом, для нормальной печати русских текстов из MS-DOS принтер должен быть русифицирован согласно CP866, а для печати из Windows - согласно CP1251. Однако мне не встречались принтеры, русифицированные в соответствии с кодировкой CP1251, все русифицированные принтеры рассчитаны на печать из MS-DOS. Это значит, что печатать кодами из Windows невозмож-

но, печать по-русски в среде Windows всегда происходит в графическом режиме, то драйверы Windows непосредственно управляют печатающей головкой принтера с помощью специальной системы команд (лазерные принтеры - приятное исключение). Далее описаны возможные способы русификации принтеров в порядке убывания их предпочтительности.

Аппаратно русифицированный принтер

Вообще говоря, только такая русификация и может считаться настоящей и полной. Принтер содержит "прошитые" русские буквы в необходимой кодировке (обычно CP866). Важно, чтобы буквы имелись всех стандартных размеров и, желательно, нескольких начертаний. Об этих дополнительных требованиях см. ниже.

Русифицированный в другой кодировке

Все еще попадаются принтеры, имеющие "прошитые" русские шрифты, но кодировка которых отличается от CP866. Такие принтеры можно использовать, если достать или написать самому простой перекодировщик, в частности, таким образом можно печатать на многих АЦПУ, не рассчитанных изначально на подключение к персональным компьютерам. Принтер позволяет загрузить шрифты в специальный буфер.

Ну что же... такой вариант лучше, чем ничего. Конечно, после каждого выключения принтера шрифт придется перегружать, но с точки зрения программы печать будет по прежнему происходить в кодах, а для принтера что "прошитые" шрифты, что загруженные - разница небольшая. На скорости это не сказывается, только доставляет некоторые неудобства пользователю. Для печати на таком принтере необходимо иметь файл со шрифтами и, в некоторых случаях, специальную программу для их загрузки.

Такой способ русификации можно использовать для старых моделей матричных принтеров, для всех лазерных принтеров и для многих струйных принтеров. К сожалению, не все струйные принтеры позволяют загружать шрифт, а лазерные имеют свою специфику. В частности, недостаточно загру-

зить шрифты в лазерный принтер, а нужно выбрать текущий, потому что за один раз можно загрузить несколько шрифтов.

Нерусифицированный без возможности загрузки шрифтов

Если принтер нерусифицирован аппаратно и не имеет буфера для загрузки русских шрифтов, то он никаким образом не позволит печатать по-русски кодами. Единственный вариант печати на таком принтере - в графическом режиме. А нерусифицированное АЦПУ можно просто выбросить (если у вас нет барабана с русскими буквами).

Этот вариант самый неприятный для программистов. Хотя... Если вы подыщете подходящий резидентный графический драйвер, то и с таким принтером можно жить в MS-DOS. Для Windows этот вариант ничего необычного не представляет, как вы помните, Windows все равно печатает в графике.

3.7. Сканер

Сканеры – устройства для оцифровки и ввода в компьютер изображений с бумажных копий - это старейших вид компьютерной периферии. Современные сканеры позволяют оцифровывать изображения даже объемных предметов и диапозитивов (слайдов).

Разрешающая способность. Для сканера это основная характеристика. Измеряется она в точках на дюйм (dpi). Сканеры низшей ценовой категории обеспечат вам разрешение сканирование до 600 dpi - этого вполне хватит для распознавания текстов и коллекционирования картинок. Другое дело - профессионалы издательского дела. Вот им порой необходимо разрешение вплоть до 1200 dpi.

Разрядность сканера – следующий важный показатель, который измеряется в битах. Фактически она означает тот количество информации, которая понадобится для оцифровки каждой точки изображения. А еще - то количество цветов, которое способен распознать ваш сканер: 24 бита соответствует 16,7 миллионам цветов, 30 бит - 1 миллиарду. Конечно, реально такое ко-

личество цветов вам никогда в быту не понадобится: чувствительности человеческого глаза не хватит даже на то, чтобы отличить 16 - битный цвет от 24 - битного. Но разработчики не унимаются - на последних моделях домашних сканеров заявлена уже 48 - битная разрядность!

Виды сканеров

Ручные сканеры. Такой сканер занимает не больше места, чем книжка среднего формата, и стоит недорого. Так что, хотя ручные сканеры и обеспечивают хороший уровень качества получаемого изображения (разрешение - до 800 x 800 dpi, разрядность - 24 бит), добиться его будет весьма непросто.

Планшетный сканер. Сканер этого типа представляет собой что-то вроде большого планшета. Бумажный лист с изображением или текстом кладется на прозрачную стеклянную поверхность, под которой "снует" распознающий элемент сканера, прибор закрывается крышкой... А дальше сканер сделает все сам - так же, как работает ксерокс. Вот только на выходе получится не бумажная, а цифровая копия картинки - файл.

Есть, конечно же, еще и другие типы сканеров: листовые, протягивающие, специализированные сканеры для фотографий и слайдов и. т. д. Однако вряд ли стоит рассказывать об этих, в общем - то, не слишком массовых устройствах.

Итак, монитор, клавиатура, мышь, сканер и принтер это периферийные устройства компьютера, в то время как к комплектующим устройствам относятся и периферийные устройства, и материнская плата и другие функциональные узлы, находящиеся в системном блоке. Комплектующие устройства входят в обязательный состав компьютера.

Тема 4. Алгоритмы и алгоритмизация. Визуализация алгоритмов – 3 часа

Решение любой задачи является творческим процессом, который состоит из нескольких *последовательных этапов*. К ним относятся :

А. *Анализ постановки задачи и ее предметной области*

-понимание постановки и требований исходной задачи, определение предметной области, для которой поставлена задача,

-анализ предметной области, выявление данных, которые фиксируют входную и выходную информацию (определение их структуры и свойств),

определение отношений между данными, условий и ограничений, накладываемых на эти отношения,

Б. *Формальное моделирование решения задачи*

-выбор и применение формальной системы для описания модели предметной области и решения задачи,

-формирование основной идеи, выбор методов решения задачи,

-определение технологий, средств и исполнителя решения задачи, построение алгоритмов, реализующих выбранные методы,

В. *Практическое решение*

-применение выбранных методов и средств для решения ,

-анализ полученных результатов.

Эти этапы ориентированы для получения решения не отдельно взятой, конкретной задачи, а некоторого класса задач данного типа. Этап построения алгоритмов, реализующих выбранные методы решения задачи, детализирует и визуализирует процесс ее решения. Алгоритмизация позволяет уже на этом этапе оценить эффективность решения, уточнить методы решения для различных потоков входных данных и выявить некоторые ошибки.

В этой последовательности наиболее трудоемким и рутинным является этап применения выбранных методов и средств для решения задачи. В настоящее время наиболее распространенным средством для решения задач является ЭВМ. Применение выбранных методов и алгоритмов для решения на ЭВМ включает дальнейшую детализацию ее решения за счет описания последовательности применяемых операций в виде программы для ЭВМ. Это придает процессу решения не только визуальные качества, но и качества интерактивности.

Не все задачи, решаемые с помощью ЭВМ, требуют составления сложных программ. Например, задачи вычислений в электронных таблицах или задачи поиска и выборки данных в базах данных. Решение некоторых задач благодаря внедрению новых информационных технологий вообще не требуют программирования, что расширяет сферу применения ЭВМ. Однако, и при решении этих задач необходимо выполнение вышеприведенных этапов.

Целью данной работы является рассмотрение этапов решения задачи с использованием ЭВМ, при этом наибольшее внимание уделяется составлению алгоритмов или алгоритмизации, так как, на мой взгляд, этот этап является достаточно трудоемким и важным. Любые ошибки, возникающие при построении алгоритмов, приводят к серьезным погрешностям при решении задач.

Эта работа предназначена для тех, кто не умеет, но стремится научиться использовать ЭВМ при решении задач, составлять корректные алгоритмы и на их основе правильные программы. Умение составлять алгоритмы позволит получить детальное решение и может быть использовано при любых технологиях проектирования программ от структурного программирования до объектно-ориентированной и компонентно-ориентированной технологии.

4.1. . Анализ постановки задачи и ее предметной области

На первом этапе уточняется постановка задачи, после чего выявляются отдельные явления, объекты, процессы, их связи и зависимости предметной области.

Здесь определяются такие понятия как *исходные* и *результирующие данные*, абстрактно представляющие информацию о процессах предметной области реального мира, а также поясняются, каким образом из исходных данных могут быть получены результирующие.

Исходные данные должны быть полными, т.е. содержать данные, необходимые и достаточные для решения задачи. Если данные неполные, необходимо приложить дополнительные усилия для сбора дополнительных сведе-

ний; эта ситуация может также возникнуть на последующих этапах при выборе метода решения.

Различают исходные данные *трех видов*: постоянные, условно-постоянные и переменные.

*Постоянные исходные данные*_- это данные, которые сохраняют свои значения в процессе решения задачи (математические константы, координаты неподвижных объектов) и не зависят от внешних факторов.

Условно-постоянные данные - это данные, которые могут иногда изменять свои значения; причем эти изменения не зависят от процесса решения задачи, а определяются внешними факторами (величина подоходного налога, курс валют, количество дней в году).

Переменные данные - это данные, которые изменяют свои значения в процессе решения задачи.

На этом этапе важно не только классифицировать данные по отношению к процессу решения, но определить их наименование, тип, структуру и ограничения, накладываемые на значения. Желательно также определить допустимые и недопустимые операции по отношению к различным типам исходных данных.

На рис.6 представлена классификация данных.

Данное относят к *простому типу*, если в любой момент времени оно определяет одно и только одно значение. Диапазон изменения возможных значений определяется типом данных. Например, требуется вычислить площадь поверхности некоторого тела. Очевидно, что для представления информации о вычисляемой площади поверхности некоторого тела достаточно использовать данное простого числового типа. Простые данные определяют такое отношение: одно имя - одно значение.

Классификация данных по структурному признаку

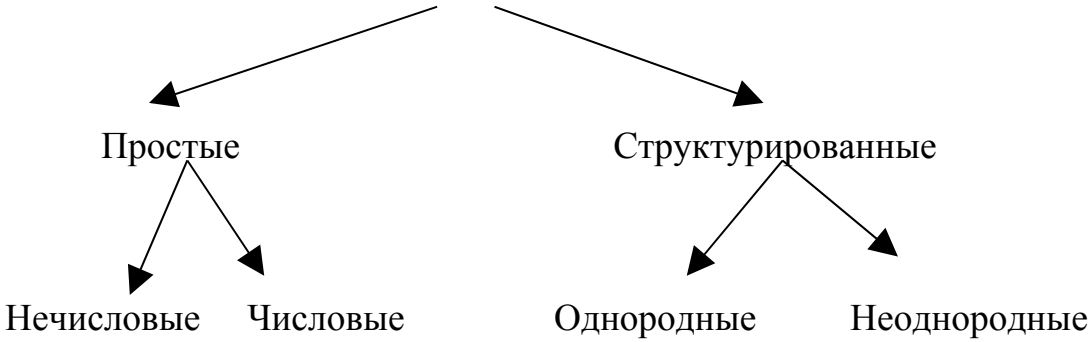


Рисунок 6 – Классификация данных

Структурированные данные отличаются от простых тем, что к ним применимо другое отношение: одно имя - много значений. Если все элементы, входящие в такую структуру, однотипны, то такая структура называется однородной, в противном случае - неоднородной. Классическим примером однородной структуры является *массив*, являющийся последовательностью однотипных значений, таких как, например, (2,51,3,7,88). Неоднородная структура в отличие от однородной содержит значения различных типов, относящихся к одному понятию или объекту, и, значит, такое структурированное данное несет в себе больше информации. Для представления неоднородных структур используют *запись*. *Запись* - это структура, предназначенная для представления данных различного типа. Запись состоит из поименованных полей, каждое из которых должно содержать значение определенного типа.

Рассмотрим простой пример. Задача заключается в определении в некоторой стране города с максимальным количеством жителей. Данные, которые необходимо проанализировать, это нечисловые данные, содержащие информацию о названии города, и числовые данные, содержащие информацию о численности населения в этом городе. В качестве структуры, содержащей данные о названии города и количестве в нем жителей, следует выбрать неоднородную структуру - запись, пример которой изображен в таблице 6.

Таблица 6

Пример записи

Имя поля: *Город*	Имя поля: *Количество жителей*
Тип поля: *Строка символов*	Тип поля: *Число*
Значение: *Москва*	Значение: *8 578 676*

В качестве структуры, содержащей информацию о множестве городов рассматриваемой страны, можно выбрать однородную структуру типа массив, состоящий из записей таблицы 6.

Определение отношений между данными, условий и ограничений, накладываемых на значения данных и эти отношения, зависит от конкретной постановки задачи и требований пользователя.

В результате анализа постановка и требования задачи могут быть представлены в обобщенном виде.

4.2. Формальное моделирование решения задачи

После проведения анализа постановки задачи, выявления данных, их структуры и отношений между ними можно приступить к построению формальной модели. Это наиболее важный этап в процессе решения задачи.

Модель – это упрощенное представление о реальном объекте, процессе или явлении.

Моделирование – построение моделей для исследования и изучения моделируемого объекта, процесса, явления с целью получения новой информации при решении конкретных задач.

Для описания модели предметной области решаемой задачи необходимо выбрать некоторую формальную систему. Обычно, исходя из постановки задачи, можно сразу определить один или несколько видов моделей, подходящих для описания и моделирования решения вашей задачи: математические, геометрические, структурные, логические и др.

Наиболее распространенными и хорошо изученными являются математические модели, описывающие зависимости между данными числового типа. Например, в качестве математической модели звезды можно использовать систему уравнений, описывающих процессы, происходящие в недрах звезды. Математической моделью другого рода являются математические соотношения, позволяющие рассчитать оптимальный план работы предприятия. К основным достоинствам математических моделей безусловно относят-

ся хорошо изученные и широко применяемые математические методы решения большого класса задач, что значительно облегчает формирование основной идеи и выбор методов решения задачи.

Приступая к разработке модели, следует попытаться решить задачу для конкретных входных данных, затем обобщить полученное решение на основе его анализа для любых значений входных данных. Перед тем как определить решение задачи для конкретных входных данных целесообразно найти ответ на следующие вопросы:

Существуют ли решения аналогичных задач?

Какая математическая модель больше всего подходит для решения этой задачи?

Пример:

Постановка задачи. Требуется определить пригодность данной аудитории для проведения учебных занятий.

Решение.

Этап 1. Анализ постановки задачи и ее предметной области.

В результате анализа предметной области, выявляем, что эта предметная область связана с образовательным процессом. И постановка задачи может быть переформулирована следующим образом. Определить, подходит ли некоторая аудитория для проведения занятий группы учеников при некоторой норме площади для каждого ученика. Введем обозначения для входных и выходных данных. Исходные данные должны быть представлены простыми переменными значениями числового типа: A - ширина аудитории, B - ее длина, K - количество учеников в группе, N - допустимое минимальное количество квадратных метров для одного ученика (норма), M - количество парт в аудитории. В качестве выходных данных будут выступать сообщения: " Аудитория может быть использована для поведения учебных занятий " и " Аудитория не может быть использована для поведения учебных занятий ".

Этап 2. Формальное решение.

Определим отношения между входными и выходными данными. Для этого введем промежуточные данные числового типа : S - площадь аудитории, C - требуемая по нормам площадь для проведения занятий для группы из K учеников, D - требуемое количество парт для обучения группы, состоящей из K учеников. Опишем соотношения между входными и выходными данными, используя математические зависимости.

Математическая модель:

$$S = A*B,$$

$C=N*K$, $S>=C$, $K<=2*D$.

4.3. Основы алгоритмизации

Слово «*алгоритм*» появилось в 9-м веке и связано с именем математика Аль-Хорезми, который сформулировал правила выполнения четырех арифметических действий над многозначными числами.

В настоящее время понятие алгоритма - одно из фундаментальных понятий науки информатика. С одной стороны алгоритм является предметом изучения такой отрасли математики как теория алгоритмов (Марков [1]), с другой стороны в информатике существует неформальное определение алгоритма, и алгоритмизация выступает в качестве общего метода информатики.

Объектом приложения алгоритмов являются самые различные науки и области практической деятельности (Хохлюк[3],Ахо [2]). Широкое применение алгоритмов для решения практических задач не только при использовании ЭВМ позволяет рассматривать эту область информатики как отдельную дисциплину - *алгоритмику*.

Алгоритм – это точно определенная последовательность действий для некоторого исполнителя, выполняемых по строго определенным правилам и приводящих через некоторое количество шагов к решению задачи.

Исполнитель алгоритмов определяет элементарные действия, из которых формируется алгоритм. Отдельные действия, составляющие алгоритм, называются операциями. При этом под операцией понимается как какое-то

единичное действие, например, сложение, так и группа взаимосвязанных действий.

Основными особенностями любого алгоритма являются решение задачи в обобщенном виде и возможность выполнять действия по решению задачи для конкретных значений (не только человеку, но и различным техническим устройствам (исполнителям)). Основным исполнителем несложных алгоритмов является человек. Достаточно вспомнить последовательность действий для решения систем линейных уравнений, вычисления корней уравнений.

При решении сложных задач исполнителем является ЭВМ и составление алгоритма решения задачи является необходимым этапом, детализирующим метод решения для дальнейшего программирования. Программа осуществляет еще более глубокую детализацию решения и его визуализацию.

Свойства алгоритма:

Определенность – выполнив очередное действие, исполнитель должен точно знать, что ему делать дальше.

Дискретность – прежде, чем выполнить определенное действие, надо выполнить предыдущее.

Массовость – по одному и тому же алгоритму решаются однотипные задачи и неоднократно.

Понятность – алгоритм строится для конкретного исполнителя человеком и должен быть ему понятен. Это облегчает его проверку и модификацию при необходимости .

Результативность – алгоритм всегда должен приводить к результату.

Можно сказать, что в процессе формального решения задачи, ее решение сначала описывается на языке математики в виде системы формул, а затем на языке алгоритмов в виде некоторого процесса, в котором используются ранее определенные математические формулы и условия их выполнения.

Таким образом, алгоритм может рассматриваться как связующее звено в цепочке "метод решения - реализующая программа".

4.4. Основные средства представления алгоритмов

Алгоритм моделирует решение задачи в виде точно определенной последовательности действий для некоторого исполнителя по преобразованию исходных данных в результирующие. Процесс составления алгоритмов называют *алгоритмизацией*.

Алгоритм, реализующий решение задачи, можно представить различными способами: с помощью *графического* или *текстового* описания, в виде *таблицы значений*.

Графический способ представления алгоритмов имеет ряд преимуществ благодаря визуальности и явному отображению процесса решения задачи. Алгоритмы, представленные графическими средствами, получили название *визуальные алгоритмы*.

Текстовое описание алгоритма является достаточно компактным и может быть реализовано на абстрактном или реальном языке программирования в виде программы для ЭВМ.

Таблицы значений представляют алгоритм неявно, как некоторое преобразование конкретных исходных данных в выходные. Табличный способ описания алгоритмов может быть с успехом применен для проверки правильности функционирования разработанного алгоритма на конкретных тестовых наборах входных данных, которые вместе с результатами выполнения алгоритма фиксируются в "таблицах трассировки".

Таким образом, все три способа представления алгоритмов можно считать взаимодополняющими друг друга. На этапе проектирования алгоритмов наилучшим способом является графическое представление, на этапе проверки алгоритма - табличное описание, на этапе применения - текстовая запись в виде программы.

4.5. Визуальные алгоритмы

При проектировании визуальных алгоритмов используют специальные графические элементы, называемые *графически блоками*, которые представлены на рис. 7. Результатом алгоритмизации решения задачи является блок-схема алгоритма, состоящая из некоторой последовательности таких графических блоков.

Общими правилами при проектировании визуальных алгоритмов являются следующие:

3 В начале алгоритма должны быть блоки ввода значений входных данных.

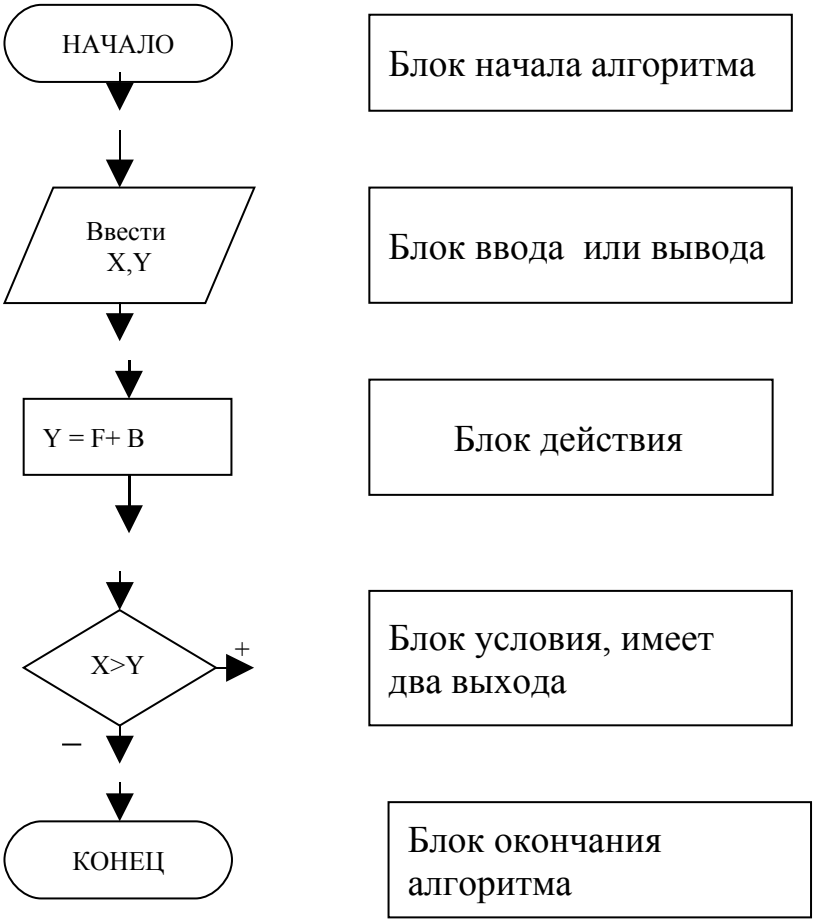
4 После ввода значений входных данных могут следовать блоки обработки и блоки условия.

5 В конце алгоритма должны располагаться блоки вывода значений выходных данных.

6 В алгоритме должен быть только один блок начала и один блок окончания.

7 Связи между блоками указываются направленными или ненаправленными линиями.

Рисунок 7 – Основные блоки визуальных алгоритмов



Этап проектирования алгоритма следует за этапом формального решения задачи, на котором определены входные и выходные данные, а также зависимости между ними.

При построении алгоритмов для сложной задачи используют системный подход с использованием *декомпозиции* (нисходящее проектирование сверху-вниз). Как и при разработке любой сложной системы, при построении алгоритма используют дедуктивный и индуктивный методы. При дедуктивном методе рассматривается частный случай общеизвестных алгоритмов. Индуктивный метод применяют в случае, когда не существует общих алгоритмических решений.

Одним из системных методов разработки алгоритмов является *метод структурной алгоритмизации*. Этот метод основан на визуальном представлении алгоритма в виде последовательности управляющих структурных фрагментов. Выделяют *три базовые* управляющие процессом обработки ин-

формации *структуры*: *композицию, альтернативу и итерацию*. С помощью этих структур можно описать любые процессы обработки информации.

Композиция (следование) – это линейная управляющая конструкция, не содержащая альтернативу и итерацию. Она предназначена для описания единственного процесса обработки информации.

Альтернатива – это нелинейная управляющая конструкция, не содержащая итерацию. Она предназначена для описания различных процессов обработки информации, выбор которых зависит от значений входных данных.

Итерация – это циклическая управляющая структура, которая содержит композицию и ветвление. Она предназначена для организации повторяющихся процессов обработки последовательности значений данных.

В соответствии с наличием в алгоритмах управляющих структур композиции, альтернативы и итерации *алгоритмы классифицируют* на:

-линейные,

-разветвленные,

-циклические алгоритмы.

Линейные алгоритмы не содержат блока условия. Они предназначены для представления линейных процессов. Такие алгоритмы применяют для описания обобщенного решения задачи в виде последовательности модулей. Пример линейного алгоритма приведен на рисунке 8.

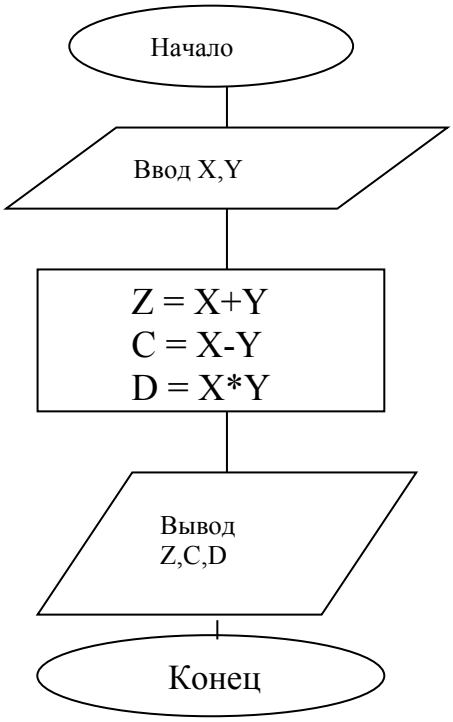
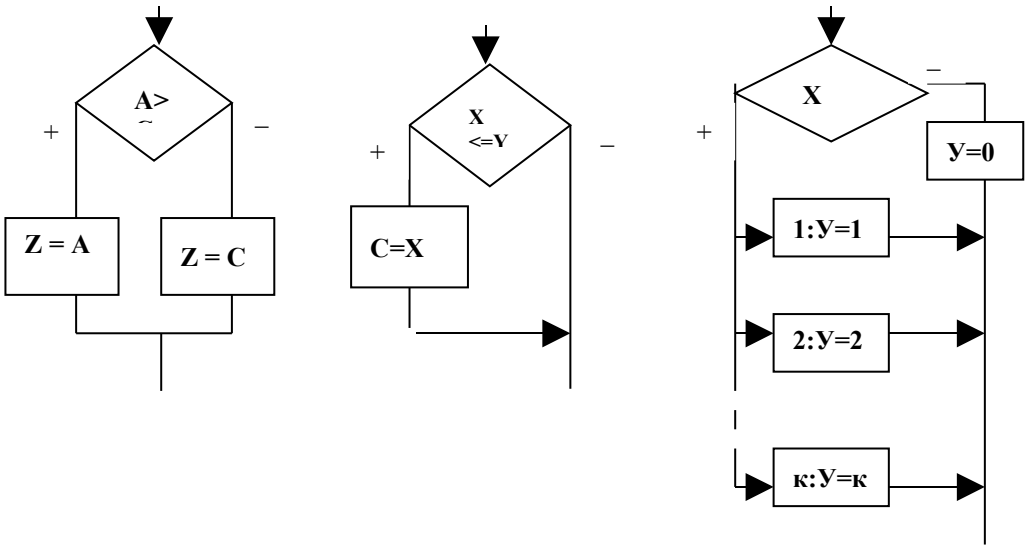


Рисунок 8 – Пример линейного визуального алгоритма

4.6. Разветвленные алгоритмы

Разветвленные алгоритмы в своем составе содержат блок условия и различные конструкции ветвления (рис. 9). *Ветвление* – это структура, обеспечивающая выбор между альтернативами.



а) ветвление б)неполное ветвление в) многоальтернативный выбор

Рисунок 9 – Структуры ветвления

Каждая управляющая структура ветвления имеет один вход и один выход. Ветвления содержат блок условия, в котором записывают логические условия, такие как A >C , X<= Y. В зависимости от значений переменных A,C в управляющей структуре ветвления на рис. 9а условие A >C принимает значение "истина" или "ложь" и процесс вычислений включает блок действия Z=A или Z=C. Аналогично происходит и в управляющей структуре *неполного ветвления* (рис. 9б). Только в этом случае , если условие X<= Y истинно, то выполняется действие C=X, в противном случае никаких действий не выполняется.

В управляющей структуре *многоальтернативный выбор* в блоке условия записывается переменная, в данном случае X, которая может принимать различные значения (рис. 9в). Если значение пременной X совпадет с одним из значений в блоке действия, то выполняется действия , записанные в этом блоке. Например, если X=1, то выполнится действие У=1. Если значение X не совпало ни с одним из значений, указанных в блоках справа, то выполняется действие в блоке слева, которого также как и в неполном ветвлении может и не быть.

Пример:

Составить алгоритм нахождения минимального значения из 3-х чисел. *Решение.* Для определения минимального значения будем использовать проверку пары значений. Визуальные разветвленные алгоритмы приведены на рис. 10, 11, 12. Эти алгоритмы использует для обозначения чисел переменные значения A,B,C и вложенные структуры ветвления.

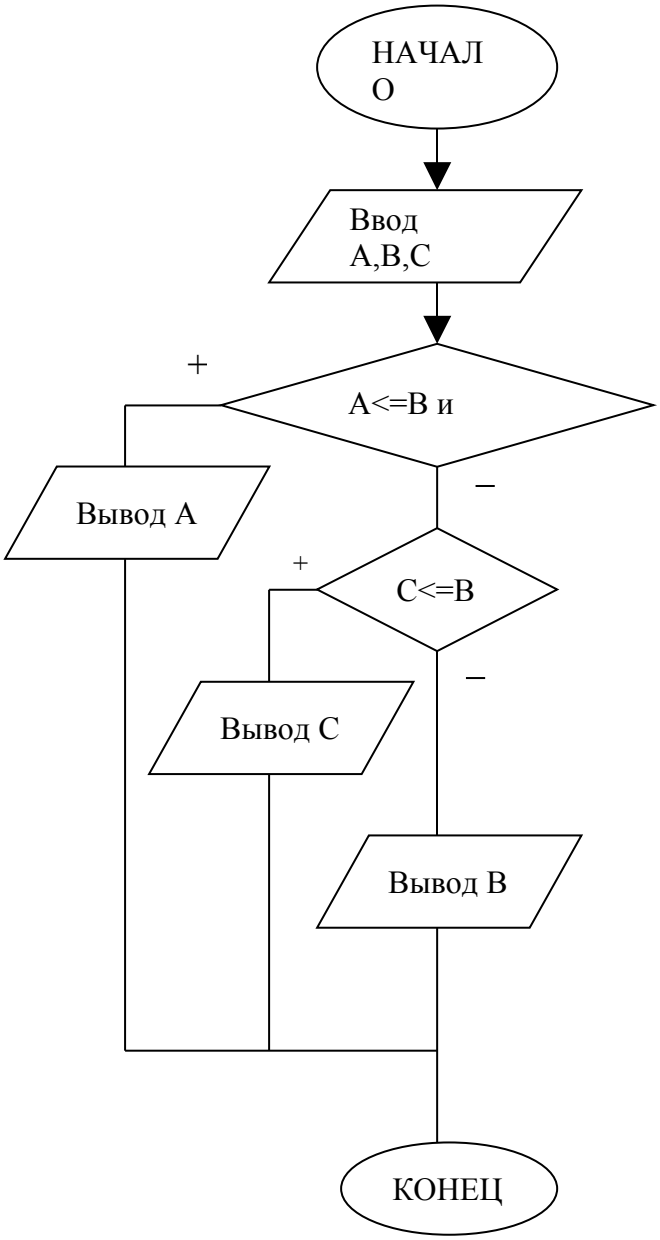


Рисунок 10 – Поиск минимального значения из трех чисел A,B,C при помощи двойного сравнения.

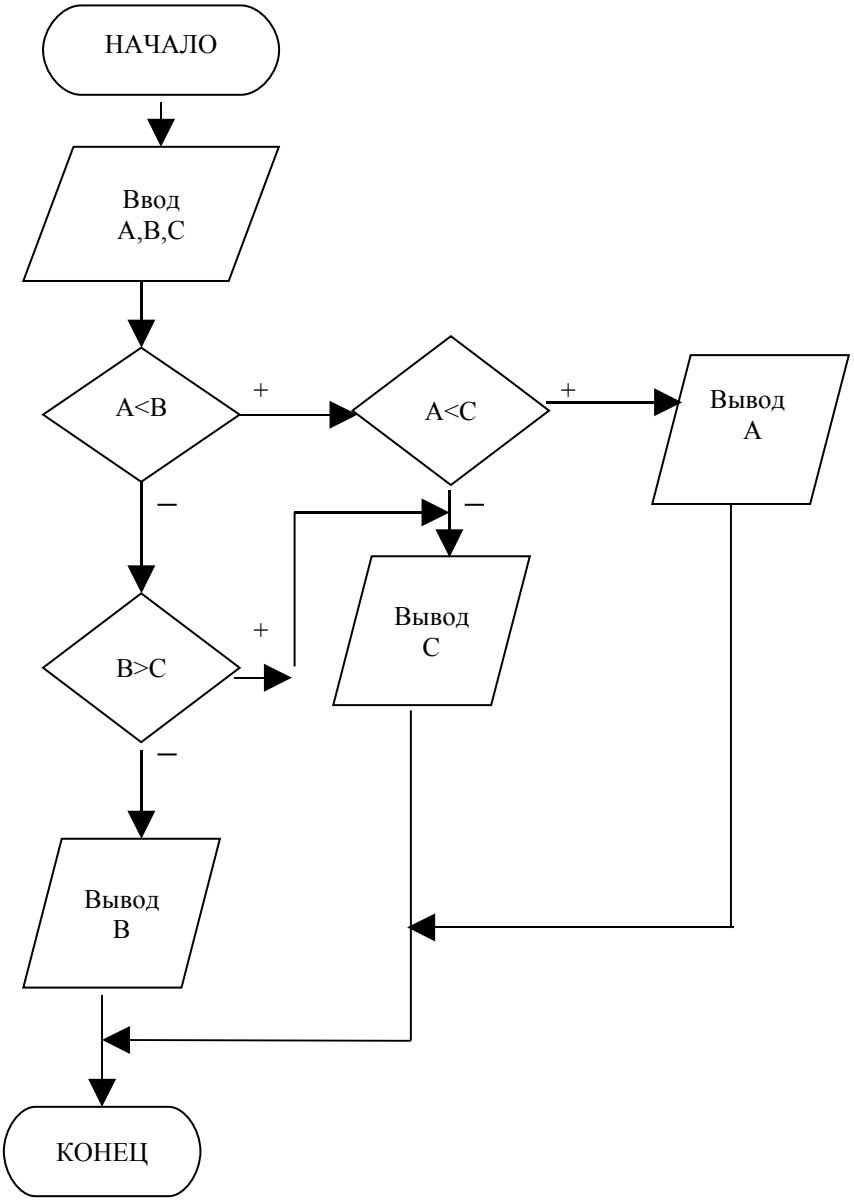


Рисунок 11 – Поиск минимального числа из трёх A,B,C. Метод последовательного сравнения .

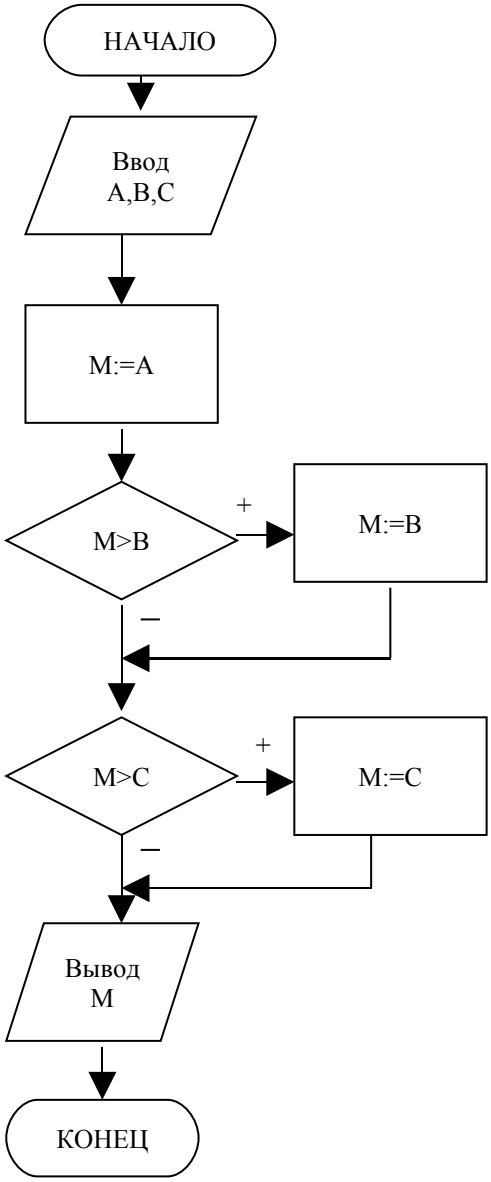


Рисунок 12 – Поиск минимального числа из трёх A,B,C.
Метод сравнения с промежуточной переменной M.

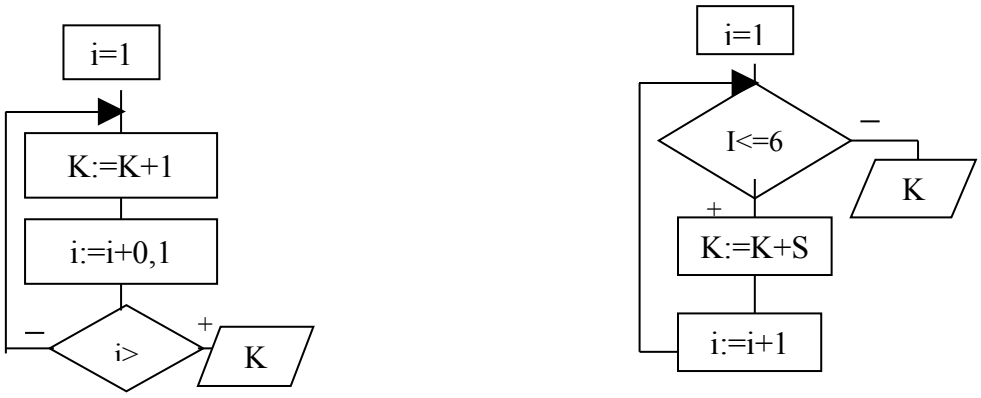
4.7. Циклические алгоритмы

Циклические алгоритмы являются наиболее распространенным видом алгоритмов, в них предусматривается повторное выполнение определенного набора действий при выполнении некоторого условия. Такое повторное выполнение часто называют циклом.

Существуют два основных видов циклических алгоритмов: циклические алгоритмы с предусловием, циклические алгоритмы с постусловием. Они отличаются друг от друга местоположением условия выхода их цикла.

Цикл с предусловием начинается с проверки условия выхода из цикла. Это логическое выражение, например I<=6. Если оно истинно, то выполняются те действия, которые должны повторяться. В противном случае, если логическое выражение I<=6 ложно, то этот цикл прекращает свои действия.

Цикл с постусловием функционирует иначе. Сначала выполняется один раз те действия, которые подлежат повторению, затем проверяется логическое выражение , определяющее условие выхода из цикла, например, I>6 .Проверка его осуществляется тоже по-другому. Если условие выхода истинно, то цикл с постусловием прекращает свою работу, в противном случае - происходит повторение действий, указанных в цикле. Повторяющиеся действия в цикле называются "*телом цикла*". Разновидности циклов приведены на рис. 13а,б.



а) Цикл с постусловием б) Цикл с предусловием

Рисунок 13 – Виды циклических алгоритмов

Классическим примером циклического алгоритма служит алгоритм для вычисления степени числа $Y=X^n$. Этот алгоритм может быть реализован на основе операции умножения. Табличное представление такого алгоритма, отражающего зависимость У от Х при изменении показателя степени n от 1 до 3, представлено в табл.7. В этой таблице показаны также рекуррентные соотношения между У и Х, определяющие как на каждом шаге зависит значение У от значения Х и от значения У, вычисленного на предыдущем шаге.

Таблица 7

Рекуррентные соотношения при вычислении $Y=X^n$

n	Y	Реккурентные соотношения
1	Y[1]=X	Y=X
2	Y[2]=X*X или Y[2]=Y[1]*X	Y=X*X или Y=Y*X
3	Y[3]=X*X*X или Y[3]=Y[2]*X	Y=X*X*X или Y=Y*X

Пример:

Пусть требуется составить алгоритм вычисления суммы ряда

$S=x+x^2+x^3+\ldots+x^n$.

Решение. Исходные данные для алгоритма это переменные x и n. На каждом шаге будем вычислять очередной член суммы Y и прибавлять его к предыдущему значению суммы S.Для этого используем рекуррентную формулу вычисления степени Х (см. таблицу 3) Y=Y*X, тогда сумма ряда на каждом шаге итерации будет вычисляться по формуле S=S+Y. Количество итераций К изменяется от 1 до n и равно количеству членов ряда. Начальное значение суммы ряда S равно 0. На рис. 14 представлен циклический алгоритм с предусловием для вычисления заданной суммы ряда.

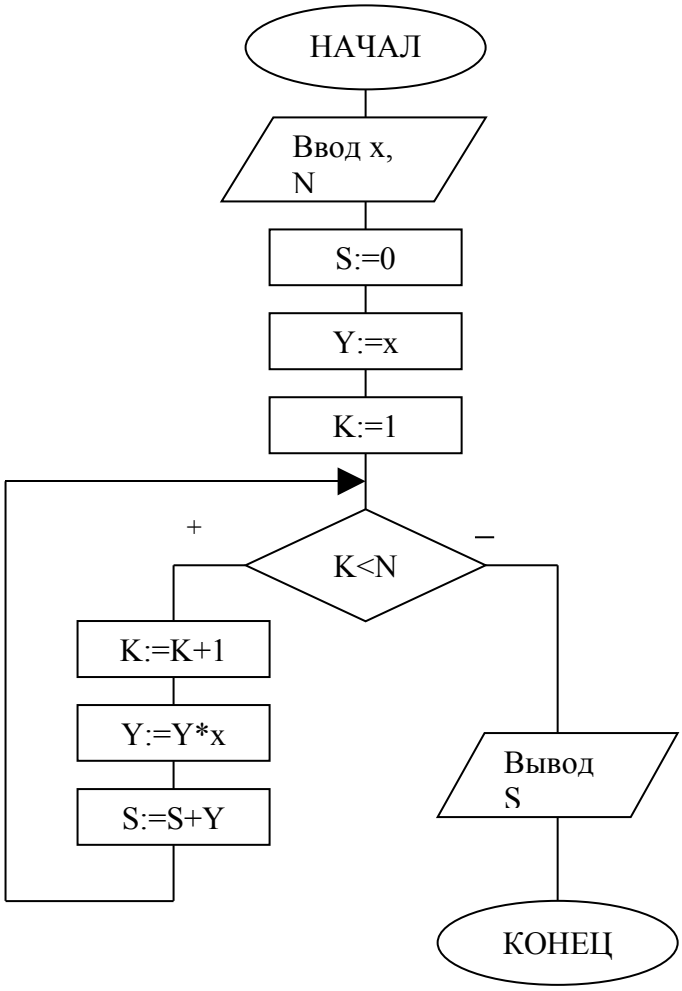


Рисунок 14 – Алгоритм вычисления суммы ряда $S=x+x^2+x^3+\ldots+x^n$

Пример:

Требуется составить алгоритм получения на отрезке [-15,15] множества значений функции Y= SIN(X) в виде таблицы значений (X,Y) при изменении аргумента X по формуле X[k]=X[k-1]+h, где h=1,5.

Решение. Такие задачи относят к задачам табулирования функций. Из условия задачи определяем, что начальное значение отрезка табулирования X= -15, конечное значение - X=15. Процесс получения множества пар X,Y) является итерационным, значит проектируемый алгоритм будет циклическим. Условие выхода из цикла X>15. На рис. 15 представлен циклический алгоритм с предусловием вычисления табличного значения функции Y= SIN(X) на отрезке -15<X<15 при изменении X на каждом шаге итерации на величину 1,5. Результатом выполнения алгоритма является циклический вывод множеств пар (Y,X).

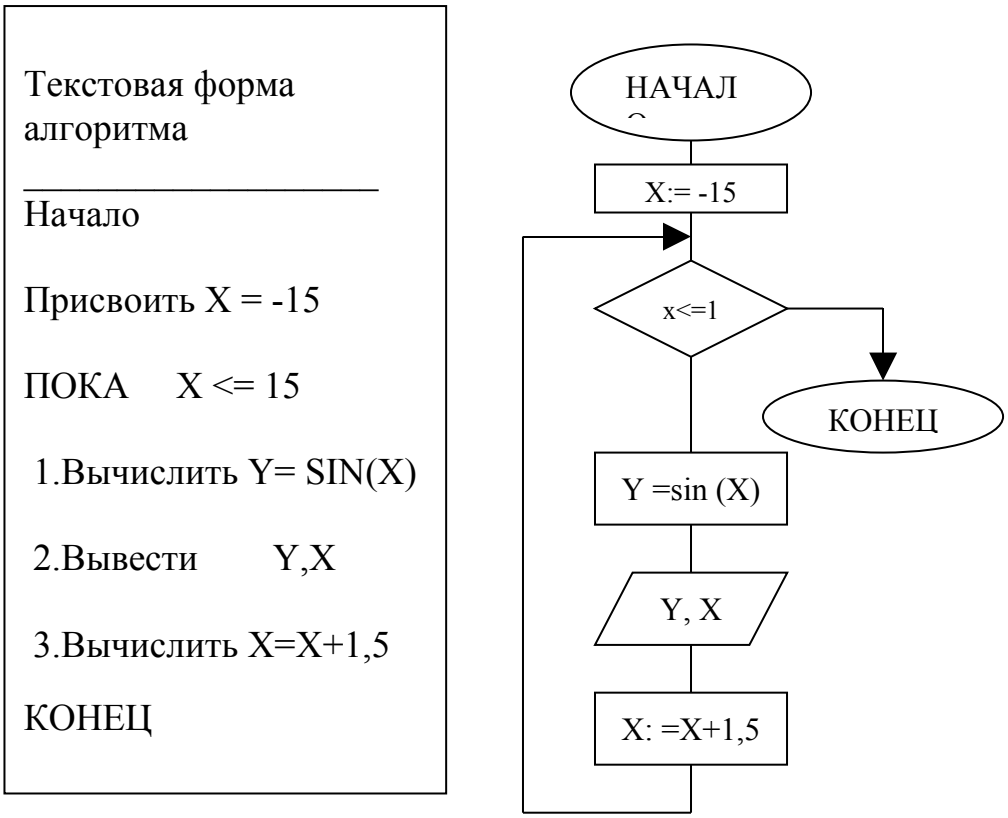


Рисунок 15 – Циклический алгоритм табулирования функции Y =sin (X)

4.8. Алгоритмы обработки последовательностей чисел

Последовательность значений – это набор однотипных величин, которые вводятся и обрабатываются циклически. Примером последовательности целых чисел может быть следующий набор значений: (2,5,-4,10,1,0). Последовательности значений отличаются от массивов значений тем, что в памяти одновременно все значения последовательности не хранятся. Для обозначения значения последовательности используют одну переменную, в которую на каждом шаге итерации вводится очередное значение последовательности. Отличительной особенностью последовательности является также возможность содержания неопределенного или неизвестного заранее количества ее значений. В этом случае критерием окончания последовательности служит некоторое особое значение, например, ноль.

Пример:

В числовой последовательности определить сумму положительных и произведение отрицательных чисел. Решение представить с использованием циклического алгоритма с предусловием. Признак конца последовательности - значение 0.

Решение. Обозначим за X переменную, содержащую очередное значение последовательности, за S - сумму положительных значений , за P - произведение отрицательных значений. Полученный алгоритм приведен на рис. 16.

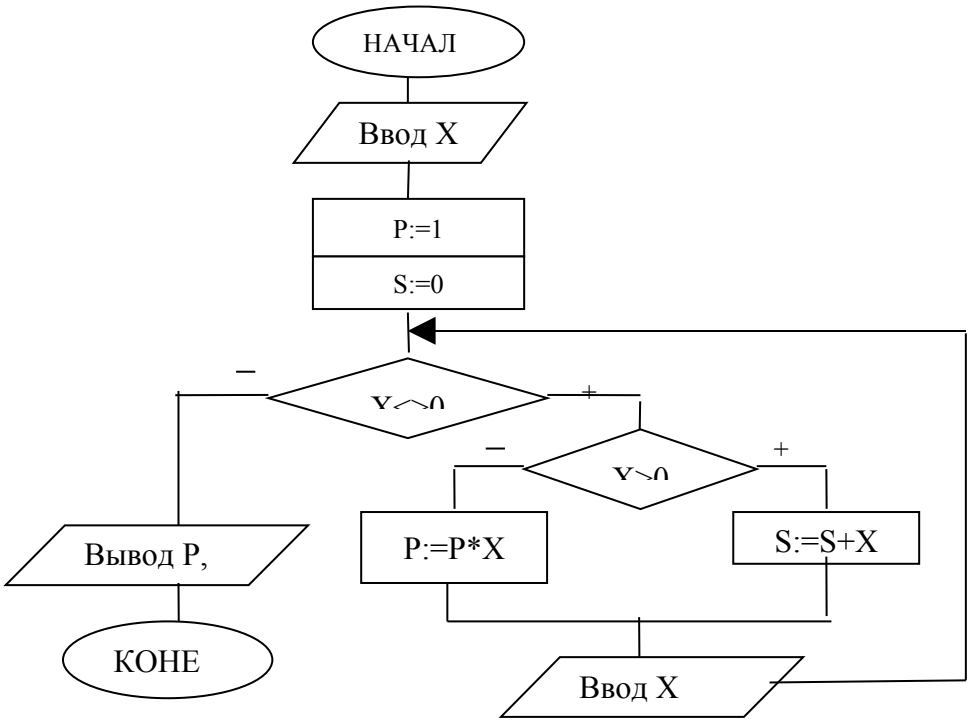


Рисунок 16 – Алгоритм вычисления суммы положительных и произведения отрицательных значений числовой последовательности

Условие для выбора вычислений X>0.

Для вычисления суммы значений воспользуемся рекуррентной формулой S=S+X с начальным значением S=0, для вычисления произведения - рекуррентной формулой P=P*X с начальным значением P=1.

Условие выхода из цикла неравенство X<>0.

4.9. Алгоритмы обработки одномерных числовых массивов

Под структурой данных типа *массив* понимают однородную структуру однотипных данных, одновременно хранящихся в последовательных ячейках оперативной памяти. Эта структура должна иметь имя и определять заданное количество данных (элементов). Однотипность данных определяет возможность использования циклических алгоритмов для обработки всех элементов массива. Количество итераций цикла определяется количеством элементов массива. Одновременное хранение в памяти всех элементов массива позволяет решать большой набор задач, таких как, поиск элементов, упорядочение и изменение порядка следования элементов.

Доступ к любому элементу массива осуществляется по его номеру (индексу). Поэтому для обращения к элементу массива используют имя_массива(номер элемента), например, A(5).

Массив называется одномерным , если для получения доступа к его элементам достаточно одной индексной переменной.

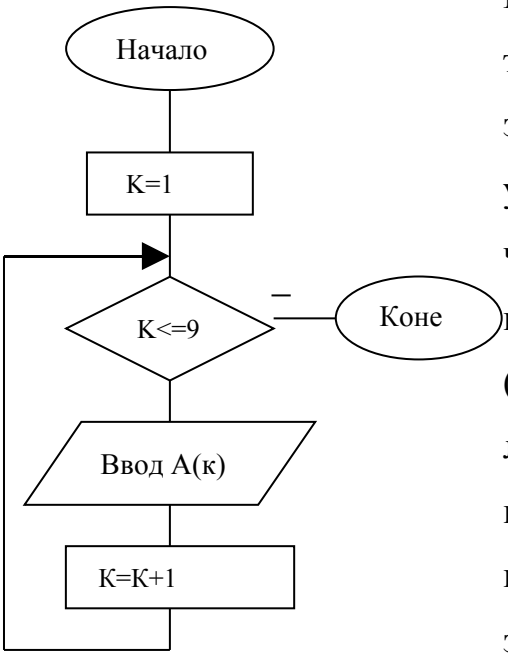


Рисунок 17 – Алгоритм ввода элементов

Рассмотрим простой алгоритм ввода элементов одномерного числового массива A из 9 элементов. В этом циклическом алгоритме условие выхода из цикла определяется значением специальной переменной К, которая называется счетчиком элементов массива A (рис.17), эта же переменная К определяет количество итераций циклического алгоритма ввода элементов массива. На каждом шаге итерации переменная К (значение номера элемента массива A) изменяется на 1, то есть происходит переход к новому элементу массива. В дальнейшем,

при рассмотрении алгоритмов обработки одномерных массивов в целях устранения дублирования алгоритм ввода элементов массива будем заменять одним блоком, подразумевая, что он реализуется по схеме, циклического алгоритма, представленного на рисунке 17.

Пример:

Составить алгоритм определения в одномерном числовом массиве A из N элементов суммы положительных элементов.

Решение. Алгоритм представлен на рисунке 18. В этом алгоритме перемен-

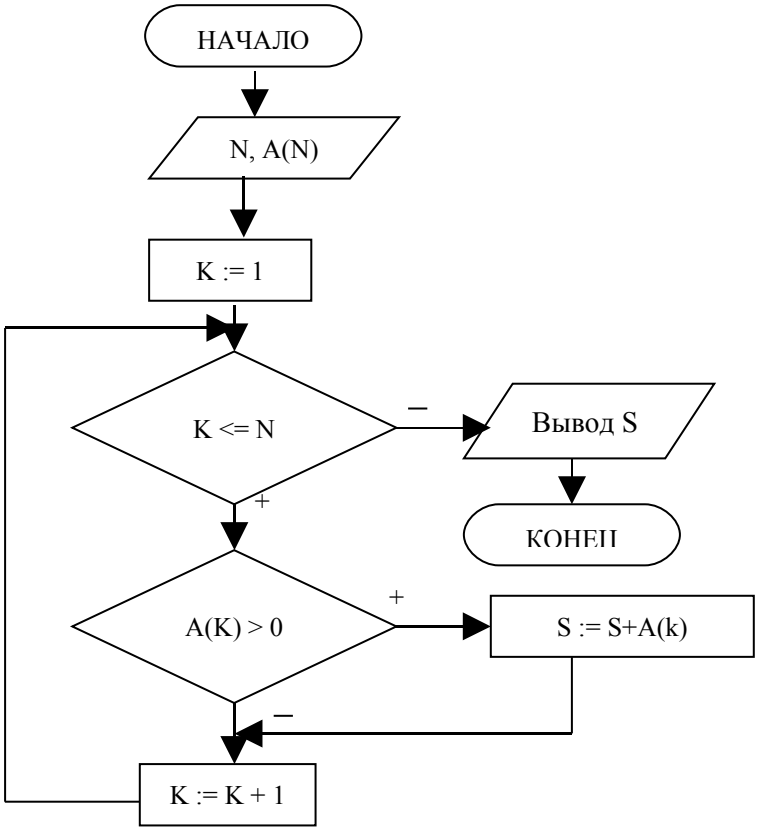


Рисунок 18 – Алгоритм вычисления суммы положительных элементов

ная K - является счетчиком элементов массива, S - сумма элементов массива, она вычисляется по рекуррентной формуле S=S+A(K). Ввод количества и значений элементов массива осуществляется вначале в отдельном блоке ввода, который реализуется по схеме алгоритма ввода элементов массива, изображенного на рис.17.

Часто для проверки правильности работы алгоритмов на конкретных наборах данных используют таблицу трассировки. Эта таблица содержит столько столбцов, сколько переменных и условий в алгоритме, в ней мы выполняем действия шаг за шагом от начала до конца алгоритма для конкретных наборов входных данных.

Пример:

Составить алгоритм поиска элемента с максимальным значением в одномерном массиве A(1..n) и его таблицу трассировки для значений (3, 7, 0, 9).

Решение. Введем обозначения K- текущий номер элемента, A[K] - текущее значение элемента массива, N=4 количество элементов одномерного массива, M- номер максимального элемента массива, A[M] - значение максимального элемента массива. Основной идеей алгоритма является выполнение сравнения текущего элемента массива A[K] и элемента с максимальным значением A[M], определенным на предыдущем шаге итерации. По алгоритму изображенному на рис.19 получено максимальное значение для массива (3, 7, 0, 9), процесс и правильный результат поиска которого показаны в таблице 8.

Рисунок 19 – Алгоритм поиска максимального значения в массиве

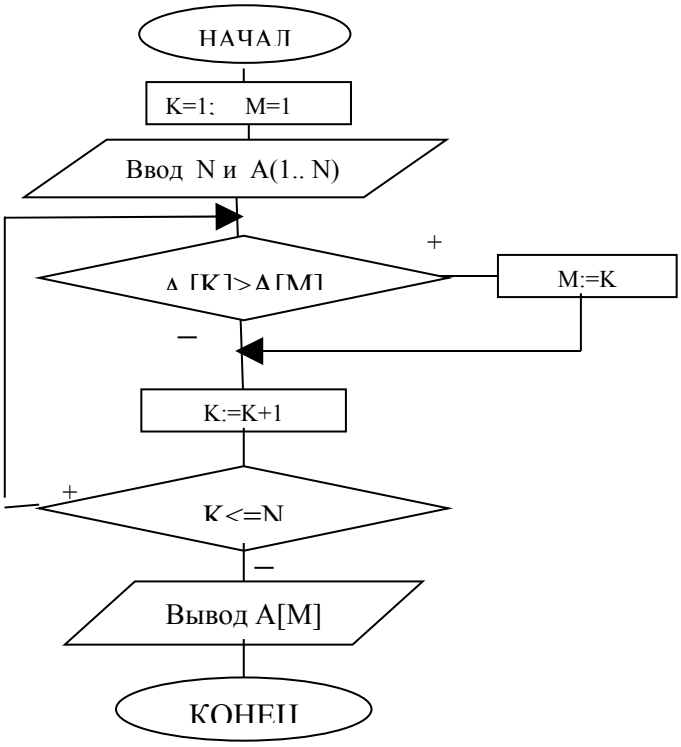


Таблица 8

Таблица трассировки алгоритма примера

Номер элемента массива K	Значение элемента A (K)	Номер максимального M		Значение максимальнго A(M)		Проверка A(K)>A(M)
1	3	1		3		нет
2	7	1	2	3	7	да
3	0	2		7		нет
4	9	2	**4**	7	**9**	да

4.10. Алгоритмы обработки двумерных массивов

Двумерный массив – это структура однотипных элементов, расположенных в виде таблицы значений. Такое представление значений соответствует математическому понятию двумерный массив. Каждый элемент в двумерном массиве идентифицируется *номером строки* и *номером столбца*, на пересечении которых он расположен. Например, в двумерном массиве A, изображенном на рис. 20, элемент со значением 5 расположен на пересечении третьей строки и второго столбца. Этот элемент будет обозначаться как A(3,2). А элемент A(1,4) имеет значение , равное нулю. Такое представление набора значений позволяет выполнять обработку как отдельных значений в двумерном массиве, так и последовательности значений, расположенных в строках или столбцах.

9	7	1	0
12	2	6	1
10	5	12	4
6	22	31	3

Рисунок 20 – Пример двумерного массива

В дальнейшем будем считать, что для двумерного массива A(N,M) в обозначении элемента A(i,j) первое значение i соответствует номеру строки и изменяется от1 до N, а j - номеру столбца и изменяется от 1 до M. В отличие от одномерного массива, в котором использовался только один номер для определения местоположения элемента и требовался только один цикл для ввода элементов, в двумерном массиве для обработки элементов необходимы два вложенных друг в друга цикла. Внешний цикл предназначен для изменения номера строки i, а второй, внутренний, - для изменения номера столбца j в текущей строке i.
На рис. 21 представлен простой алгоритм ввода элементов, построенный в виде структуры из вложенных циклов.

При рассмотрении в дальнейшем алгоритмов обработки элементов двумерного массива в целях сокращения их размера фрагмент ввода элементов будем заменять отдельным блоком ввода.

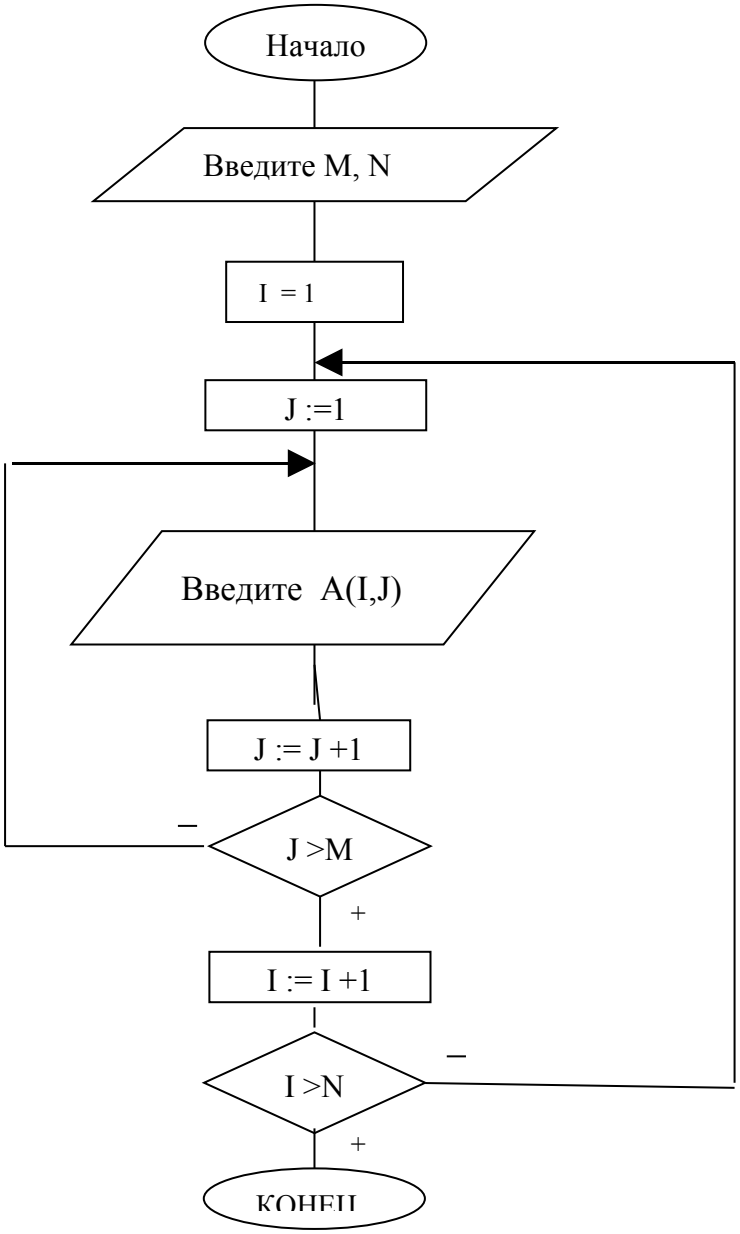


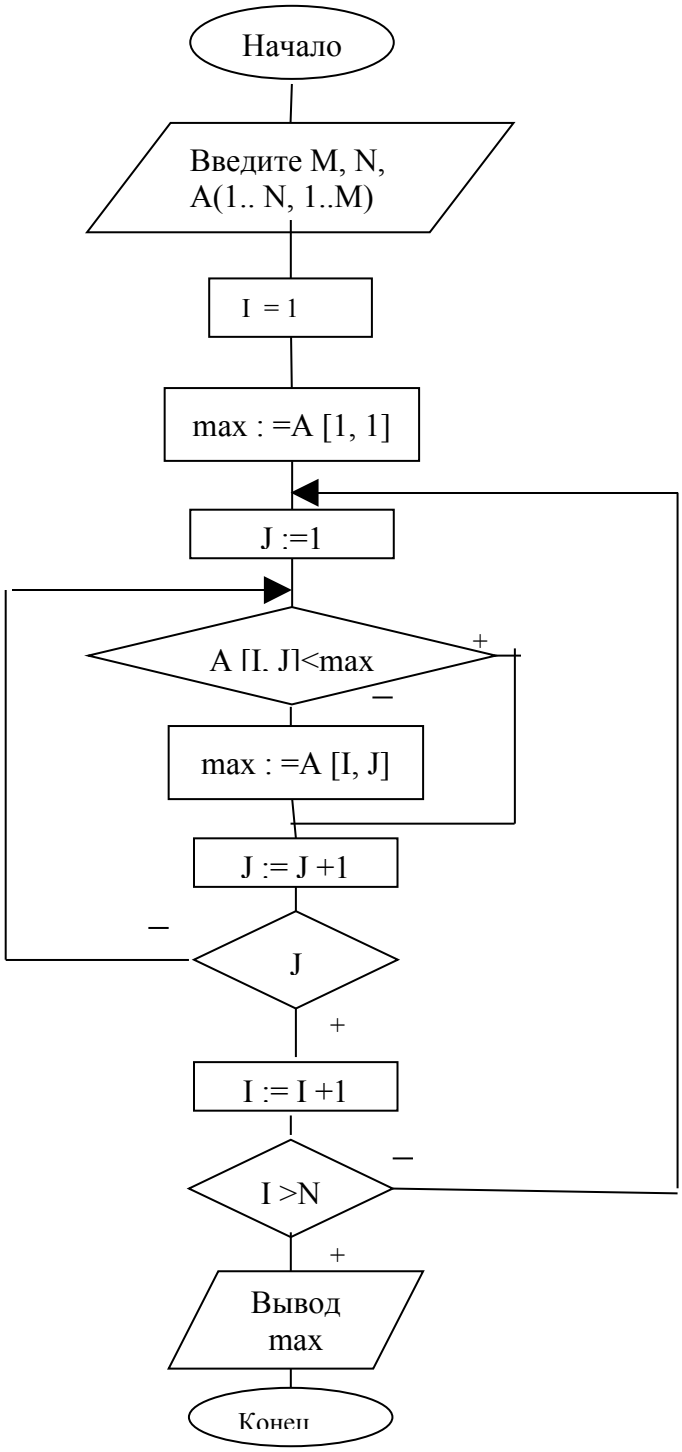
Рисунок 21 – Алгоритм ввода элементов двумерного массива

Пример:

Составить алгоритм поиска максимального значения в двумерном массиве A(N,M).

Решение. Поиск максимального элемента в двумерном массиве осуществляется аналогично поиску в одномерном массиве. Отличие состоит в том, что для обработки двумерного массива используем вложенные циклы. Обозначим максимальный элемент переменной MAX. Значение этой переменной будет меняться на каждой итерации цикла, если очередной значение элемента массива окажется больше MAX (см. рис. 22).

Рисунок 22 – Алгоритм поиска максимального значения в двумерном массиве



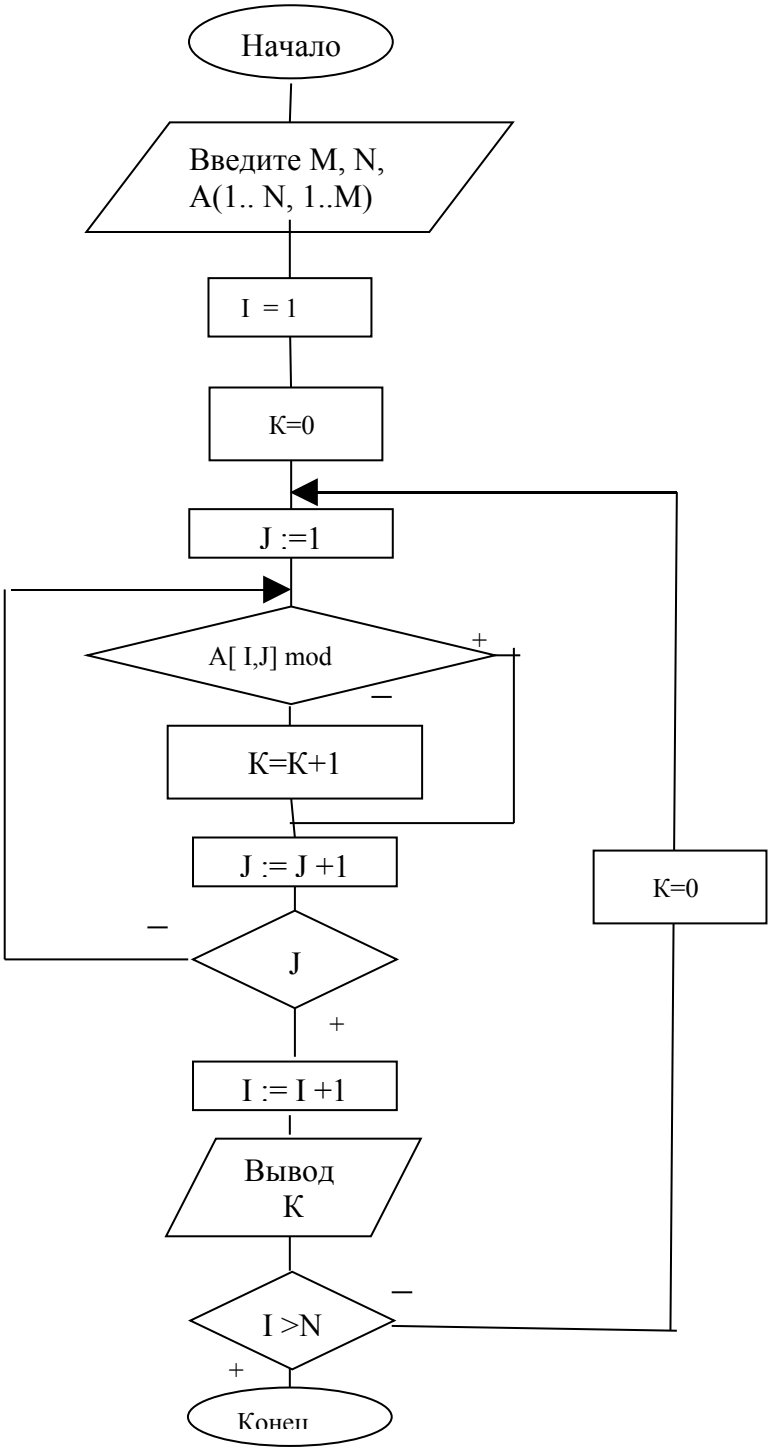
Пример:

Составить алгоритм вычисления количества нечетных элементов в каждой строке двумерного массива A(1.. N, 1..M).

Решение. Для определения нечетных элементов будем использовать проверку на нечетность A[I,J] mod 2 ≠ 0, для подсчета количества нечетных

значений - формулу К=К+1и вывод значения К столько раз, сколько строк в массиве. Алгоритм решения представлен на рис. 23.

Рисунок 23 – Алгоритм вычисления в каждой строке двумерного мас-



сива количества нечетных элементов

Тема 5. Программирование – 2 часа

5.1. Компьютер как исполнитель алгоритмов.

Под *алгоритмом* понимают постоянное и точное предписание (указание) исполнителю совершить определенную последовательность действий, направленных на достижение указанной цели или решение поставленной задачи.

Для программиста, составляющего программы на универсальных языках программирования компьютер является универсальным исполнителем. Иначе говоря, на таких языках можно составить программу решения любой задачи по обработке информации.

Программы составляются программистами по алгоритмам для решения определенных задач. Человек не должен объяснять исполнителю свои цели и смысл команд программы. Очевидно, что компьютер и не сможет понять смысла совершаемых им действий. Кроме того, компьютер не обладает способностью к анализу результатов, например, с точки зрения их соответствия постановке задачи. Компьютер не может обойтись без программы и исходных данных, подготовить которые под силу только человеку. С этой точки зрения решение задачи компьютером — это формальное исполнение алгоритма ее решения, закодированного и хранимого вместе с данными в оперативной памяти.

Для человека этот факт важен потому, что он должен понимать ограниченность возможностей компьютера как исполнителя, необходимость самому предусмотреть все тонкости команд, поручаемых компьютеру для исполнения, и что вся ответственность за использование компьютеров обществом лежит только на людях.

Пользователь (не программист) работает с каким-либо средством прикладного программного обеспечения (текстовым редактором, табличным процессором, бухгалтерским пакетом программ и т. п.). В этом случае компьютер для него является специализированным исполнителем, ориентированным на определенный тип работы (редактирование текста, табличные расчеты, вычисление заработной платы и пр.). Такой пользователь может и

не знать, какие услуги компьютера реализуются аппаратными, а какие — программными средствами. Для него компьютер является «виртуальной машиной», обслуживающей его информационные потребности. Аппаратное обеспечение (hardware) и программное обеспечение (software) компьютера для пользователя представляются единым целым.

5.2. Программа как изображение алгоритма в терминах команд, управляющих работой компьютера

Компьютер (ЭВМ) — это универсальное (многофункциональное) электронное программно-управляемое устройство для хранения, обработки и передачи информации.

Компьютер является универсальным исполнителем по обработке информации. Следовательно, для него, как для любого исполнителя, свойственна определенная система кодирования информации. Эта система команд называется языком машинного кодирования. Состав языка машинных команд был предложен Джоном фон Нейманом еще в 1946 г. и во многом сохранился в современных компьютерах.

Программа управления компьютером — это последовательность команд языка машинных команд.

Каждая *команда* является директивой для процессора на выполнение определенного действия. Эти действия выполняет либо сам процессор (арифметические и логические операции), либо внешние устройства (команды ввода, вывода) под управлением процессора.

Согласно принципам Дж. фон Неймана, программа во время ее исполнения и обрабатываемые ею данные находятся в оперативной памяти ЭВМ (принцип хранимой в памяти программы). И то и другое имеет вид двоичных кодов. Процессор исполняет программу, начиная с первой команды и заканчивая на последней (или на специальной команде «стоп»). Во время исполнения очередной команды процессор переписывает ее в свои регистры, исполняет и переходит к следующей команде.

5.3. Коды, ассемблеры, языки высокого уровня.

Программисты, работавшие на ЭВМ первого поколения (ламповые машины 50—60 х гг.), писали программы на *ЯМК* (Языки машинных команд). Это довольно сложная работа. Для облегчения программирования созданы *языка программирования высокого уровня*.

Примерами таких языков являются Фортран, Паскаль, Бейсик, Си и др. Составлены программ на таких языках много проще, чем на ЯМК. Программирование стало доступно большему числу людей.

Однако процессор ЭВМ понимает только ЯМК и не понимает ЯПВУ (Языки программирования высокого уровня). Поэтому, для того чтобы на машине исполнилась программа, записанная, например, на Паскале, ее нужно перевести с языка Паскаль на ЯМК. Такой перевод называется трансляцией. Трансляцию выполняет этот же самый компьютер по специальной программе, которая называется транслятором (например, транслятор с Паскаля). Программы-трансляторы составляются системными программистами и входят в программное обеспечение компьютера.

Языки программирования принято делить на *пять поколений*. В первое поколение входят языки, созданные в начале 50-х годов, когда первые компьютеры только появились на свет. Это был *первый язык ассемблера*, созданный по принципу “одна инструкция – одна строка”.

Расцвет второго поколения языков программирования пришёлся на конец 50-х – начало 60-х. Тогда был разработан *символический ассемблер*, в котором появилось понятие переменной. Он стал первым полноценным языком программирования. Благодаря его возникновению заметно возросли скорость разработки и надёжность программ.

Коды ассемблера

Написание ассемблерных программ знаний организации всей системы компьютера. В основе компьютера лежат понятия бита и байта. Они являются тем средством, благодаря которым компьютерной памяти представлены данные и команды.

Программа в машинном коде состоит из различных сегментов для определения данных, для машинных команд и для сегмента, названного стеком, для хранения адресов. Для выполнения арифметических действий, пересылки данных и адресации компьютер имеет ряд регистров.

Цель данного примера – проиллюстрировать простую программу на машинном языке, ее представление в памяти и результаты ее выполнения.

Программа показана в шестнадцатеричном формате:

Таблица 9

Пример машинных кодов: непосредственные данные

Команда	Назначение
B82301	Переслать шестнадцатеричное значение 0123 в AX.
052500	Прибавить шестнадцатеричное значение 0025 к AX.
8BD8	Переслать содержимое AX в BX.
03D8	Прибавить содержимое AX к BX.
8BCB	Переслать содержимое BX в CX.
2BC8	Вычесть содержимое AX из AX (очистка AX).
90	Нет операции.

Можно заметить, что машинные команды имеют различную длину: один, два или три байта. Машинные команды находятся в памяти непосредственно друг за другом. Выполнение программы начинается с первой команды, и далее последовательно выполняются остальные.

Языки программирования высокого уровня существенно отличаются от машинно-ориентированных (низкого уровня) языков. Во-первых, машинная программа, в конечном счете, записывается с помощью лишь двух символов 0 и 1. Во-вторых, каждая ЭВМ имеет ограниченный набор машинных операций, ориентированных на структуру процессора. Как правило, этот набор состоит из сравнительно небольшого числа простейших операций, типа: пере-

слать число в ячейку; считать число их ячейки; увеличить содержимое ячейки на +1 и т.п. Команда на машинном языке содержит очень ограниченный объем информации, поэтому она обычно определяет простейший обмен содержимого ячеек памяти, элементарные арифметические и логические операции. Команда содержит код и адреса ячеек, с содержимым которых выполняется закодированное действие.

Языки программирования, имитирующие естественные языки, обладающие укрупненными командами, ориентированными на решение прикладных содержательных задач, называются *языками высокого уровня*.

Языки программирования высокого уровня имеют следующие *достоинства*:

-алфавит языка значительно шире машинного, что делает его гораздо более выразительным и существенно повышает наглядность и понятность текста;

-набор операций, допустимых для использования, не зависит от набора машинных операций, а выбирается из соображений удобства формулирования алгоритмов решения задач определенного класса;

-конструкции команд (операторов) отражают содержательные виды обработки данных и задаются в удобном для человека виде;

-используется аппарат переменных и действий с ними;

-поддерживается широкий набор типов данных.

Таким образом, языки программирования высокого уровня являются машинно-независимыми и требуют использования соответствующих программ-переводчиков (трансляторов) для представления программы на языке машины, на которой она будет исполняться. Примеры языков высокого уровня.

Fortran

Это первый компилируемый язык созданный Джимом Бэкусом в 50-е годы. Программисты, разрабатывавшие программы исключительно на ассем-

блере, выражали серьезное сомнение в возможности появления высокопроизводительного языка высокого уровня, поэтому основным критерием при разработке компиляторов Фортрана являлась эффективность исполняемого хода. Хотя в Фортране был впервые реализован ряд важнейших понятий программирования, удобство создания программ было принесено в жертву возможности получения эффективного машинного кода. Однако для этого языка было создано огромное количество библиотек ,начиная от статических комплексов и кончая пакетами управления спутниками, поэтому Фортран продолжает активно использоваться во многих организациях, а сейчас ведутся работы над очередным стандартом Фортрана F2k,который появился в 2000 году. Имеется стандартная версия Фортрана HPF (High Perfomance Fortran) для параллельных суперкомпьютеров со множеством процессоров

Cobol

Это компилируемый язык для применения в экономической области и решения бизнес задач, разработанный в начале 60-х годов. Он отличается большой "многословностью" -его операторы выглядят как обычные английские фразы. В Коболе были реализованы очень мощные средства работы с большими объемами данных ,хранящимися на различных внешних носителях. На этом языке создано очень много различных приложений, которые активно эксплуатируются и сегодня. Достаточно сказать, что наибольшую зарплату в США получают программисты на Коболе.

Algol

Компилируемый язык, созданный в 1960 году. Он был призван заменить Фортран, но из-за более сложной структуры не получил широкого распространения. В 1968 году была создана версия Алгол68,по своим возможностям опережающая и сегодня многие языки программирования ,однако из-за отсутствия достаточно эффективных компьютеров для нее не удалось своевременно создать хорошие компиляторы.

Pascal

Язык Паскаль, созданный в конце 70-х годов основоположником множества идей современного программирования Никлаусом Виртом, во многом напоминает Алгол, но в нем ужесточен ряд требований к структуре программы и имеются возможности, позволяющие успешно применять его при создании крупных проектов.

Basic

Для этого языка имеются и компиляторы и интерпретаторы, а по популярности он занимает первое место в мире. Он создавался в конце 60-х годов в качестве учебного пособия и очень прост в изучении.

C

Данный язык был создан в лаборатории Bell и первоначально не рассматривался как массовый. Он планировался для замены ассемблера, чтобы иметь возможность создавать столь же эффективные и компактные программы, и в то же время не зависеть от конкретного вида процессора.

C++

C++ – это объектно-ориентированное расширения языка Си, созданное Бьярном Страуструпом в 1980 году. Множество новых мощных возможностей, позволивших резко увеличить производительность программистов, наложилось на унаследованную от языка Си определенную низкоуровневость, в результате чего создание сложных и надежных программ потребовало от разработчиков высокого уровня профессиональной подготовки.

Java

Этот язык был создан компанией Sun в начале 60-х годов на основе Си++. Он призван упростить разработку приложений на основе Си++ путем исключения из него всех низкоуровневых возможностей. Но главная особенность этого языка -компиляция не в машинный код, а в платформо-независимый байт-код. Этот байт-код может выполнятся с помощью интерпретатора виртуальной машины Java-машины JVM (Java Virtyal Machine), версии которой созданы сегодня для любых платформ. Благодаря наличию Java-машин программы на Java можно переносить на не только на уровне исходных тек-

стов, но и на уровне обычного байт-кода, поэтому по популярности язык Java сегодня занимает первое второе с мире после Бейсика.

5.4. Трансляция и компоновка

С помощью языка программирования создаётся не готовая программа, а только её текст, описывающий ранее разработанный алгоритм. Чтобы получить *работающую программу*, надо этот текст либо автоматически перевести в машинный код (для этого служат программы *компиляторы*) и затем использовать отдельно от исходного текста, либо сразу выполнять команды языка, указанные в тексте программы (этим занимаются программы-*интерпретаторы*).

Интерпретатор берёт очередной оператор языка из текста программы, анализирует его структуру и затем сразу исполняет (обычно после анализа оператор транслируется в некоторое промежуточное представление или даже машинный код для более эффективного дальнейшего исполнения). Только после того как текущий оператор успешно выполнен, интерпретатор перейдёт к следующему. При этом если один и тот же оператор будет выполняться в программе многократно, интерпретатор будет выполнять его так как, как будто встретил впервые. Вследствие этого программы, в которых требуется осуществить большой объём вычислений, будут выполняться медленно. Кроме того, для выполнения программы на другом компьютере там тоже должен стоять интерпретатор – ведь без него текст является просто набором символов.

По-другому можно сказать, что интерпретатор моделирует некоторую вычислительную виртуальную машину, для которой базовыми инструкциями служат не элементарные команды процессора, а операторы языка программирования.

Компиляторы полностью обрабатывают весь текст программы (он иногда называется исходный код) Они просматривают его в поиске синтаксических ошибок (иногда несколько раз), производят определенный смысловой

анализ, а затем автоматически переводят (транслируют) на машинный язык - генерируют машинный код. Нередко при этом выполняется оптимизация с помощью набора методов позволяющих повысить быстродействие программы (например, с помощью инструкций, ориентированных на конкретный процессор, путём исключения ненужных команд, промежуточных вычислений и т.д.). В результате законченная программа получается законченной и эффективной, работает

В сотни раз быстрее программы, выполняемой с помощью интерпретатора, может быть перенесена на другие компьютеры с процессором, поддерживающим соответствующий машинный код.

Недостаток компилятора – трудоёмкость трансляции языков программирования, ориентированных на обработку данных сложных структур, часто заранее неизвестной или динамически меняющейся во время работы программы. Тогда в машинный код приходиться вставлять множество дополнительных проверок, анализировать наличие ресурсов операционной системы, динамически их захватывать и освобождать, формировать и обрабатывать в памяти компьютера сложные объекты, что на уровне жестко заданных машинных инструкций осуществить довольно трудно, а для задачи почти невозможно.

С помощью интерпретатора, наоборот, допустимо в любой момент остановить программу, исследовать содержимое памяти, организовать диалог с пользователем, выполнить сколь угодно сложные преобразования и при этом постоянно контролировать состояние окружающей программно - аппаратной среды, благодаря чему достигается высокая надёжность работы. Интерпретатор при выполнении каждого оператора проверяет множество характеристик операционной системы и при необходимости максимально подробно информирует разработчика о возникающих проблемах. Кроме того, интерпретатор очень удобен для использования в качестве инструмента изучения программирования, так как позволяет понять принципы работы любого отдельного оператора языка.

В реальных системах программирования перемешаны технологии и компиляции и интерпретации. В процессе отладки программа может выполняться по шагам, а результирующий код не обязательно будет машинным – он даже может быть исходным кодом, написанном на другом языке программирования (это существенно упрощает процесс трансляции, но требует компилятора для конкретного языка), или промежуточным машинно-независимым кодом абстрактного процессора, который в различных машинных архитектурах станет выполнять с помощью интерпретатора или компилировать в соответствующий машинный код.

Тема 6. Программное обеспечение – 3 часа

Программное обеспечение компьютера – это совокупность программ, предназначенных для выполнения различных действий. В состав программного обеспечения включают программы и необходимые для их функционирования данные. Различают системные программы, предназначенные для управления и обслуживания компьютера и несистемные – программы (приложения). Все программы состоят из совокупности операторов и данных, описанных на некотором языке программирования, и создаются с помощью инструментальных программ. Все программы хранятся в файлах в виде либо текста программы на определенном языке программирования, либо в виде исполняемой программы. В первом случае для выполнения программы необходимо наличие транслятора или соответствующей системы программирования. Во втором случае, для выполнения программы достаточно просто запустить ее. Программное обеспечение принято классифицировать на три группы:

-системное;

-прикладное;

-инструментальное.

6.1. Системное программное обеспечение

В состав компьютера входит большое число функциональных элементов, таких как оперативная память, процессор, контроллеры, внешние запоминающие устройства, периферийные устройства и др. Для эффективного управления работой этими устройствами как системой используют программы, получившие название системными или системное программное обеспечение. Без системного программного обеспечения работа на компьютере невозможна.

6.1.1. Операционные системы

Основой системного программного обеспечения является *операционная система* (ОС), предназначенная для управления аппаратными и программными ресурсами компьютера, а также для организации взаимодействия (интерфейса)пользователя с компьютером. Операционная система представляет собой набор программ, хранимых в виде файлов на диске. Она автоматически загружается в оперативную память при включении и остается там до выключения компьютера. Эта операция загрузки выполняется загрузчиком-программой, которая вызывается базовой системой ввода-вывода (BIOS). BIOS размещается в постоянном запоминающем устройстве (ПЗУ), к которому доступ пользователя запрещен. Кроме вызова программы загрузчика BIOS также выполняет тестирование основных аппаратных компонентов. BIOS иногда относят к аппаратным средствам, иногда к программным.

В зависимости от аппаратных ресурсов компьютера различают *однозадачные и многозадачные ОС*, ОС с текстовым и графическим интерфейсом. К однозадачным ОС с текстовым интерфейсом относят ОС MS DOS, к многозадачным с графическим интерфейсом - UNIX, WINDOWS 98/NT. *Многозадачные ОС* управляют, распределяют ресурсы компьютера и обеспечивают:

-возможность одновременной или поочередной работы нескольких программ,

-возможность обмена данными между программами

-возможность совместного использования ресурсов компьютера несколькими программами.

Основные функции операционных систем заключаются в обеспечении удобного взаимодействия пользователя с аппаратным и программным обеспечением компьютера:

-организация и управление файловой системой

-установка, удаление, запуск программ

-обслуживание аппаратных ресурсов компьютера (с помощью дополнительных программ)

Примером часто используемых дополнительных программ по обслуживанию файловой системы и аппаратных ресурсов компьютера являются *программы архивации файлов.*

6.1.2. Организация файловой системы

Все современные дисковые операционные системы обеспечивают создание *файловой системы*, предназначенной для хранения данных на дисках и обеспечения доступа к ним.

Принцип организации файловой системы — табличный. Поверхность жесткого диска рассматривается как трехмерная матрица, измерениями которой являются номера поверхности, цилиндра и сектора. Под цилиндром понимается совокупность всех дорожек, принадлежащих разным поверхностям и находящихся на равном удалении от оси вращения. Данные о том, в каком месте диска записан тот или иной файл, хранятся в системной области диска в специальных таблицах размещения файлов (FAT-таблицах). Поскольку нарушение FAT-таблицы приводит к невозможности воспользоваться данными, записанными на диске, к ней предъявляются особые требования надежности, и она существует в двух экземплярах, идентичность которых регулярно контролируется средствами операционной системы.

Наименьшей физической единицей хранения данных является *сектор*. Размер сектора равен 512 байт. Поскольку размер FAT-таблицы ограничен, то для дисков, размер которых превышает 32 Мбайт, обеспечить адресацию к каждому отдельному сектору не представляется возможным. В связи с этим группы секторов условно объединяются в кластеры. Кластер является наименьшей единицей адресации к данным. Размер кластера, в отличие от размера сектора, не фиксирован и зависит от емкости диска.

Операционные системы MS-DOS, OS/2, Windows 95 и Windows NT реализуют 16-разрядные поля в таблицах размещения файлов. Такая файловая система называется FAT16. Она позволяет разместить в FA T-таблицах не более 65 536 записей (216) о местоположении единиц хранения данных и, соответственно, для дисков объемом от 1 до 2 Гбайт длина кластера составляет 32 Кбайт (64 сектора). Это не вполне рациональный расход рабочего пространства, поскольку любой файл (даже очень маленький) полностью оккупирует весь кластер, которому соответствует только одна адресная запись в таблице размещения файлов. Даже если файл достаточно велик и располагается в нескольких кластерах, все равно в его конце образуется некий остаток, нерационально расходующий целый кластер.

Для современных жестких дисков потери, связанные с неэффективностью файловой системы, весьма значительны и могут составлять от 25% до 40% полной емкости диска, в зависимости от среднего размера хранящихся файлов. С дисками же размером более 2 Гбайт файловая система FAT16 вообще работать не может.

В настоящее время только операционная система Windows 98 обеспечивает более совершенную организацию файловой системы — FAT32 с 32-разрядными полями в таблице размещения файлов. Для дисков размером до 8 Гбайт эта система обеспечивает размер кластера 4 Кбайт (8 секторов).

Обслуживание файловой структуры

Несмотря на то, что данные о местоположении файлов хранятся в табличной структуре, пользователю они представляются в виде иерархической

структуры — людям так удобнее, а все необходимые преобразования берет на себя операционная система. К функции обслуживания файловой структуры относятся следующие операции, происходящие под управлением операционной системы:

• создание файлов и присвоение им имен;

• создание каталогов (папок) и присвоение им имен;

• переименование файлов и каталогов (папок);

• копирование и перемещение файлов между дисками компьютера и между каталогами (папками) одного диска;

• удаление файлов и каталогов (папок);

• навигация по файловой структуре с целью доступа к заданному файлу, каталогу (папке);

Создание и именование файлов

Файл — это именованная последовательность байтов произвольной длины. Поскольку из этого определения вытекает, что файл может иметь нулевую длину, то фактически создание файла состоит в присвоении ему имени и регистрации его в файловой системе — это одна из функций операционной системы. Даже когда мы создаем файл, работая в какой-то прикладной программе, в общем случае для этой операции привлекаются средства операционной системы.

По способам именования файлов различают “короткое” и “длинное” имя. До появления операционной системы Windows 95 общепринятым способом именования файлов на компьютерах IBM PC было соглашение 83. Согласно этому соглашению, принятому в MS-DOS, имя файла состоит из двух частей: собственно имени и расширения имени. На имя файла отводится 8 символов, а на его расширение — 3 символа. Имя от расширения отделяется точкой. Как имя, так и расширение могут включать только алфавитно-цифровые символы латинского алфавита.

Соглашение 83 не является стандартом, и потому в ряде случаев отклонения от правильной формы записи допускаются как операционной систе-

мой, так и ее приложениями. Так, например, в большинстве случаев система “не возражает” против использования некоторых специальных символов (восклицательный знак, символ подчеркивания, дефис, тильда и т. п.), а некоторые версии MS-DOS даже допускают использование в именах файлов символов русского и других алфавитов. Сегодня имена файлов, записанные в соответствии с соглашением 83, считаются “короткими”.

Основным недостатком “коротких” имен является их низкая содержательность. Далеко не всегда удается выразить несколькими символами характеристику файла, поэтому с появлением операционной системы Windows 95 было введено понятие “длинного” имени. Такое имя может содержать до 256 символов. Этого вполне достаточно для создания содержательных имен файлов. “Длинное” имя может содержать любые символы, кроме девяти специальных: \ /:*?"<> |. В имени разрешается использовать пробелы и несколько точек. Расширением имени считаются все символы, идущие после последней точки.

Наряду с “длинным” именем операционные системы Windows 95 и Windows 98 создают также и короткое имя файла — оно необходимо для возможности работы с данным файлом на рабочих местах с устаревшими операционными системами.

Особенности Windows 95 и Windows 98.

Использование “длинных” имен файлов в операционных системах Windows 95 и Windows 98 имеет ряд особенностей.

1. Если “длинное” имя файла включает пробелы, то в служебных операциях его надо заключать в кавычки. Рекомендуется не использовать пробелы, а заменять их символами подчеркивания.

2. В корневой папке диска (на верхнем уровне иерархической файловой структуры) нежелательно хранить файлы с длинными именами — в отличие от прочих папок в ней ограничено количество единиц хранения, причем чем длиннее имена, тем меньше файлов можно разместить в корневой папке.

3. Кроме ограничения на длину имени файла (256 символов) существует гораздо более жесткое ограничение на длину полного имени фата (в него входит путь доступа к файлу, начиная от вершины иерархической структуры). Полное имя не может быть длиннее 260 символов.

4. Разрешается использовать символы любых алфавитов, в том числе и русского, но если документ готовится для передачи, с заказчиком (потребителем документа) необходимо согласовать возможность воспроизведения файлов с такими именами на его оборудовании.

5. Прописные и строчные буквы не различаются операционной системой. Для нее имена Письмо.txt и письмо-txt соответствуют одному и тому же файлу. Однако символы разных регистров исправно отображаются операционной системой, и, если для наглядности надо использовать прописные буквы, это можно делать.

6. Программисты давно научились использовать расширение имени файла для передачи операционной системе, исполняющей программе или пользователю информации о том, к какому типу относятся данные, содержащиеся в файле, и о формате, в котором они записаны. В ранних операционных системах этот факт использовался мало. По существу, операционные системы MS-DOS анализировали только расширения .BAT (пакетные файлы с командами MS-DOS), .EXE, .COM (исполнимые файлы программ) и .SYS (системные файлы конфигурации). В современных операционных системах любое расширение имени файла может нести информацию для операционной системы. Системы Windows 95/98 имеют средства для регистрации свойств типов файлов по расширению их имени, поэтому во многих случаях выбор расширения имени файла не является частным делом пользователя. Приложения этих систем предлагают выбрать только основную часть имени и указать тип файла, а соответствующее расширение имени приписывают автоматически.

Создание каталогов (папок)

Каталоги (папки) — важные элементы иерархической структуры, необходимые для обеспечения удобного доступа к файлам, если файлов на носителе слишком много, то файлы объединяются в каталоги по любому общему признаку, заданному их создателем (по типу, по принадлежности, по назначению, по времени создания и т. п.). Каталоги низких уровней вкладываются в каталоги более высоких уровней и являются для них вложенными. Верхним уровнем вложенности иерархической структуры является корневой каталог диска.

Все современные операционные системы позволяют создавать каталоги. Правила присвоения имени каталогу ничем не отличаются от правил присвоения имени файлу, хотя негласно для каталогов не принято задавать расширения имен. Мы знаем, что в иерархических структурах данных адрес объекта задается маршрутом (путем доступа), ведущим от вершины структуры к объекту. При записи пути доступа к файлу, проходящего через систему вложенных каталогов, все промежуточные каталоги разделяются между собой определенным символом. Во многих операционных системах в качестве такого символа используется “\” (обратная косая черта).

Особенности Windows 95 и Windows 98. До появления операционной системы Windows 95 при описании иерархической файловой структуры использовался введенный выше термин каталог. С появлением этой системы был введен новый термин — папка. В том, что касается обслуживания файловой структуры носителя данных, эти термины равнозначны: каждому каталогу файлов на диске соответствует одноименная папка операционной системы. Основное отличие понятий папка и каталог проявляется не в организации хранения файлов, а в организации хранения объектов иной природы. Так, например, в Windows 95 и Windows 98 существуют специальные папки, представляющие собой удобные логические структуры, которым не соответствует ни один каталог диска.

Копирование и перемещение файлов

В неграфических операционных системах операции копирования и перемещения файлов выполняются вводом прямой команды в поле командной строки. При этом указывается имя команды, путь доступа к каталогу-источнику и путь доступа к каталогу-приемнику.

В графических операционных системах существуют приемы работы с устройством позиционирования, позволяющие выполнять эти команды наглядными методами.

Удаление файлов и каталогов (папок)

Средства удаления данных не менее важны для операционной системы, чем средства их создания, поскольку ни один носитель данных не обладает бесконечной емкостью. Существует как минимум три режима удаления данных: удаление, уничтожение и стирание, хотя операционные системы обеспечивают только два первых режима (режим надежного стирания данных можно обеспечить лишь специальными программными средствами).

Удаление файлов является временным. В операционных системах Windows 95 и Windows 98 оно организовано с помощью специальной папки, которая называется Корзина. При удалении файлов и папок они перемещаются в Корзину. Эта операция происходит на уровне файловой структуры операционной системы (изменяется только путь доступа к файлам). На уровне файловой системы жесткого диска ничего не происходит — файлы остаются в тех же секторах, где и были записаны. Уничтожение фатов происходит при их удалении в операционной системе MS-DOS или при очистке Корзины в операционных системах Windows 95/98. В этом случае файл полностью удаляется из файловой структуры операционной системы, но на уровне файловой системы диска с ним происходят лишь незначительные изменения. В таблице размещения файлов он помечается как удаленный, хотя физически остается там же, где и был. Это сделано для минимизации времени операции. При этом открывается возможность записи новых файлов в кластеры, помеченные как “свободные”.

Для справки укажем, что операция стирания фатов, выполняемая специальными служебными программами, состоит именно в том, чтобы заполнить якобы свободные кластеры, оставшиеся после уничтоженного файла, случайными данными. Поскольку даже после перезаписи данных их еще можно восстановить специальными аппаратными средствами (путем анализа остаточного магнитного гистерезиса), для надежного стирания файлов требуется провести не менее пяти актов случайной перезаписи в одни и те же сектора. Эта операция весьма продолжительна, и поскольку массовому потребителю она не нужна, то ее не включают в стандартные функции операционных систем.

Навигация по файловой структуре

Навигация по файловой структуре является одной из наиболее используемых функций операционной системы. Удобство этой операции часто воспринимают как удобство работы с операционной системой. В операционных системах, имеющих интерфейс командной строки, навигацию осуществляют путем ввода команд перехода с диска на диск или из каталога в каталог. В связи с крайним неудобством такой навигации, широкое применение нашли специальные служебные программы, называемые файловыми оболочками.

Как и операционные системы, файловые оболочки бывают неграфическими и графическими. Наиболее известная неграфическая файловая оболочка для MS-DOS — диспетчер файлов Norton Commander, а роль графической файловой оболочки для MS-DOS в свое время исполняли программы Windows 1.0 и Windows 2.0, которые постепенно развились до понятия операционной среды (в версиях Windows 3.x) и далее до самостоятельной операционной системы (Windows 95/98).

6.2. Прикладное программное обеспечение

Прикладное программное обеспечение используется для решения задач определенной прикладной области. В качестве примеров можно привести системы тестирования знаний, системы автоматизации бухгалтерских расчетов, системы мониторинга, системы анализа эффективности инвестиций, системы документооборота и др. Разработка таких систем выполняется в несколько этапов и осуществляется на основе инструментального программного обеспечения.

Пакеты прикладных подпрограмм – это совокупность подпрограмм, составленных на одном из языков программирования и удовлетворяющих определенным единым требованиям к структуре, организации их входов и выходов, описаниям подпрограмм и т. п.

Стандартные подпрограммы имеют единую форму обращения, что обеспечивает простоту и удобство настройки параметров подпрограммы на решение конкретной задачи. В качестве примера можно привести пакеты стандартных подпрограмм по численным математическим методам решения уравнений, вычисления интегралов, нахождения экстремумов и т. п.

6.3. Инструментальное программное обеспечение

Инструментальное программное обеспечение предназначено для создания программных продуктов общего назначения, не зависящих от предметной прикладной области. Программный продукт - это некоторый файл, содержащий информацию, полученную с помощью программы.

Программный продукт может содержать как элементы информационного обеспечения, например, массив чисел и формул, список фамилий, текст документа, базы данных так и элементы программного обеспечения, к которой относят прикладные программы, призванные сами создавать программные продукты.

Следует отметить, что оболочки для создания прикладных программ создаются также инструментальными программами и поэтому могут быть от-

несены к прикладным программам. Рассмотрим кратко назначения некоторых инструментальных программ.

Различают следующие *виды инструментальных программ*:

Текстовые редакторы

Текстовые редакторы предназначены для создания и редактирования текстовых документов. Наиболее распространенными являются MS WORD, Лексикон. *Основными функциями текстовых редакторов* являются:

-работа с фрагментами документа;

-вставка объектов созданных в других программах;

-разбивка текста документа на страницы;

-ввод и редактирование таблиц;

-ввод и редактирование формул;

-форматирование абзаца;

-автоматическое создание списков;

-автоматическое создание оглавления.

Известны десятки текстовых редакторов. Наиболее доступными являются NOTEPAD (блокнот), WORDPAD, WORD. Работа конкретного редактора текста определяется обычно функциями, назначение которых отражено в пунктах меню и в справочной системе.

Графические редакторы

Графические редакторы используют для создания и обработки изображений. Специальный раздел информатики посвящен компьютерной графике, которая основана на передовых достижениях фундаментальных и прикладных наук: математики, физики, химии, статистики и др. В зависимости от принципа вывода изображения на экран монитора различают *растровую, векторную* и *фрактальную графику* и соответствующие графические редакторы. Отдельным видом компьютерной графики является *трехмерная графика*, сочетающая векторный и растровый способ построения изображения.

Редакторы растровых изображений используют для вывода минимальной единицы изображения точку. Точка имеет параметры: цвет, признак мигания, координаты по Х и У. Поэтому для растровых изображений используют понятие разрешение, выражающее максимальное количество точек на единицу длины. Для экранного разрешения возможны следующие варианты разрешения 800*600, 1024*768, 1280*1024 и др.Наиболее распространенными редакторами растровой графики являются PAINT, Adobe Photoshop.

Редакторы векторной графики используют в качестве элементарного графического примитива линию(контур).Контур может иметь любую форму - прямой, кривой, ломаной, фигуры. Каждый контур может иметь две или более опорных точек, именуемых узлами. Элемент контура, заключенный между двумя смежными опорными точками называют сегментом контура. Контур может быть открытым и закрытым и на его основе могут быть созданы новые объекты путем применения к контурам операций группировки, комбинирования и объединения. К векторным редакторам относят CorelDraw, Adobe Illustrator и др.

Программы создания электронных презентаций

Программы создания электронных презентаций получают все большее распространение в виду возможности быстрого создания визуального представления различных документов и режимов функционирования информационных и вычислительных систем. Наиболее популярным инструментальным средством создания электронных презентаций является POWER POINT.

Тема 7. Обзор языков высокого уровня – 6 часов

7.1. Понятие языка высокого уровня

Языки программирования высокого уровня – это формальные языки специально созданные для общения человека с компьютером. Каждый язык программирования, равно как и "естественный язык" (русский, английский и

т.д.) имеет алфавит, словарный запас, свои грамматику и синтаксис, а также семантику.

Алфавит – фиксированный для данного языка набор основных символов, допускаемых для составления текста программы на этом языке.

Синтаксис – система правил, определяющих допустимые конструкции языка программирования.

Семантика – система правил однозначного толкования отдельных языковых конструкций, позволяющих воспроизвести процесс обработки данных.

При описании языка и его применении используют *понятия языка*. Понятие подразумевает некоторую синтаксическую конструкцию и определяемые ею свойства программных объектов или процесса обработки данных.

Взаимодействие синтаксических и семантических правил определяет те или иные понятия языка, например, *операторы, идентификаторы, переменные, функции и процедуры, модули* и т.д. В отличие от естественных языков правила грамматики и семантики для языков программирования, как и для всех формальных языков, должны быть явно, однозначно и четко сформулированы.

7.2. Данные и типы языка программирования Паскаль

Для того , чтобы составить программу на языке Турбо Паскаль надо знать структуру программы и правила использования в программах различных данных. Самое общее правило гласит : все данные, используемые Вами, должны быть предварительно определены, т.е. Вам необходимо задуматься о том, как обозначить каждое данное (какое дать ему имя), о характере и диапазонах изменений их значений, о требуемой памяти для их размещения и о наборе допустимых к ним операций.

Для определения данных Турбо Паскаль предлагает использовать способы объявления данных в программе посредством задания их значений или путем задания их типов - для данных изменяющих свои значения при выполнении программы.

Тип данных определяет форму машинного представления, допустимый набор значений, а также те действия, которые могут быть выполнены над этими данными При определении данных посредством задания значений , тип данных может указываться, а может и не указываться, в этом случае он определяется автоматически. Если стандартных средств для реализации необходимых операций недостаточно, то Вам следует определить дополнительные типы или операции. Это можно реализовать или в самой программе или вне вашей программы, используя модули - библиотеки объявлений.
При определении данных посредством задания значений, тип данных может указываться, а может и не указываться, в этом случае он определяется автоматически.

Если стандартных средств для реализации необходимых операций недостаточно, то Вам следует определить дополнительные типы или операции. Это можно реализовать или в самой программе или вне вашей программы, используя модули - библиотеки объявлений

В математике принято классифицировать переменные в соответствии с некоторыми важными характеристиками. Производится строгое разграничение между вещественными, комплексными и логическими переменными, между переменными, представляющими отдельные значения и множество значений и так далее.

При обработке данных на ЭВМ такая классификация еще более важна. В любом алгоритмическом языке каждая константа, переменная, выражение или функция бывают определенного типа. В языке ПАСКАЛЬ существует правило: тип явно задается в описании переменной или функции, которое предшествует их использованию.

Концепция типа языка ПАСКАЛЬ имеет следующие основные свойства:

1.любой тип данных определяет множество значений, к которому принадлежит константа, которые может принимать переменная или выражение, или вырабатывать операция или функция; тип значения, задаваемого

константой, переменной или выражением, можно определить по их виду или описанию;

2.каждая операция или функция требует аргументов фиксированного типа и выдает результат фиксированного типа. Отсюда следует, что транслятор может использовать информацию о типах для проверки вычислимости и правильности различных конструкций.

Тип определяет:

3.возможные значения переменных, констант, функций, выражений, принадлежащих к данному типу;

4.внутреннюю форму представления данных в ЭВМ;

5.операции и функции, которые могут выполняться над величинами, принадлежащими к данному типу.

Обязательное описание типа приводит к избыточности в тексте программ, но такая избыточность является важным вспомогательным средством разработки программ и рассматривается как необходимое свойство современных алгоритмических языков высокого уровня. В языке ПАСКАЛЬ существуют:

1.скалярные типы данных;

2.структурированные типы данных.

1. К *скалярным типам* относятся стандартные типы и типы, определяемые пользователем.

- *Стандартные типы* включают целые, действительные, символьный, логические и адресный типы.
- *Типы, определяемые пользователем*: перечисляемый, интервальный.

2. *Структурированные типы* имеют четыре разновидности: массивы, множества, записи и файлы.

Кроме перечисленных, TURBO PASCAL включает еще два типа - процедурный и объектный.

Из группы скалярных типов можно выделить *порядковые типы*, которые характеризуются *следующими свойствами*:

-все возможные значения порядкового типа представляют собой ограниченное упорядоченное множество;

-к любому порядковому типу может быть применена стандартная функция Ord, которая в качестве результата возвращает порядковый номер конкретного значения в данном типе;

-к любому порядковому типу могут быть применены стандартные функции Pred и Succ, которые возвращают предыдущее и последующее значения соответственно;

-к любому порядковому типу могут быть применены стандартные функции Low и High, которые возвращают наименьшее и наибольшее значения величин данного типа.

В языке ПАСКАЛЬ введены понятия *эквивалентности* и *совместимости* типов.

Два типа T1 и T2 являются *эквивалентными* (идентичными), если выполняется одно из двух условий:

1)T1 и T2 представляют собой одно и то же имя типа;

2)тип T2 описан с использованием типа T1 с помощью равенства или последовательности равенств. Например:

```
type
T1 = Integer;
T2 = T1;
T3 = T2;
```

Менее строгие ограничения определены *совместимостью* типов. Например, типы являются совместимыми, если:

-они эквивалентны;

-являются оба либо целыми, либо действительными;

-один тип - интервальный, другой - его базовый;

-оба интервальные с общим базовым;

один тип - строковый, другой - символьный.

В ТУРБО ПАСКАЛЬ ограничения на совместимость типов можно обойти помощью приведения типов. Приведение типов позволяет рассматривать одну и ту же величину в памяти ЭВМ как принадлежащую разным типам.

Для этого используется конструкция:

Имя_Типа(переменная или значение).

Например, Integer('Z') представляет собой значение кода символа 'Z' в двухбайтном представлении целого числа, а Byte(534) даст значение 22, поскольку целое число 534 имеет тип Word и занимает два байта, а тип Byte занимает один байт, и в процессе приведения старший байт будет отброшен.

7.2.1. Скалярные типы

Целые типы определяют константы, переменные и функции, значения которых реализуются множеством целых чисел, допустимых в данной ЭВМ.

В Турбо Паскале используются пять целочисленных типов данных , наиболее распространенным среди которых является тип INTEGER (см.табл.10).

Таблица 10

Целые типы данных

Тип	Диапазон значений	Требуемая память
Shortint	-128 .. 127	1 байт
Integer	-32768 .. 32767	2 байта
Longint	-2147483648 .. 2147483647	4 байта
Byte	0 .. 255	1 байт
Word	0 .. 65535	2 байта

Над целыми операндами можно выполнять следующие арифметические операции: сложение, вычитание, умножение, деление, деление на цело, получение остатка от деления.

Знаки этих операций:

+	-	*	/	div	mod

Результат арифметической операции над целыми операндами есть вели чина целого типа. Результат выполнения операции деления целых величин есть целая часть частного. Результат выполнения операции получения остатка от деления - остаток от деления целых.

Например: 17 div 2 = 8, 3 div 5 = 0. 17 mod 2 = 1, 3 mod 5 = 3.

Операции отношения, примененные к целым операндам, дают результат логического типа TRUE или FALSE (истина или ложь).

В языке ПАСКАЛЬ имеются следующие операции отношения: равенство =, неравенство <>, больше или равно >=, меньше или равно <=, больше >, меньше < .

К аргументам целого типа применимы следующие стандартные (встроенные) функции, результат выполнения которых имеет целый тип:

Abs(X), Sqr(X), Succ(X), Pred(X), и которые определяют соответственно абсолютное значение X, X в квадрате,X+1,X-1.

Следующая группа стандартных функций для аргумента целого типа дает действительный результат:

Sin(X), Cos(X), ArcTan(X), Ln(X), Exp(X), Sqrt(X).

Эти функции вычисляют синус, косинус и арктангенс угла, заданного в радианах, логарифм натуральный, экспоненту и корень квадратный соответственно.

Результат выполнения функции проверки целой величины на нечетность Odd(X) имеет значение истина, если аргумент нечетный, и значение ложь, если аргумент четный:

X=5 Odd(X)=TRUE , X=4 Odd(X)=FALSE.

Для быстрой работы с целыми числами определены процедуры:

Inc(X) X:=X+1

Inc(X,N) X:=X+N

Dec(X) X:=X-1

Dec(X,N) X:=X-N

Примеры объявления целочисленных данных:

```
var
a:integer;  c,d:byte;  f:shortint;
const
step=1; mm:word=65500;
```

Действительные (вещественные) типы данных представляют собой вещественные значения, которые используются в арифметических выражениях и занимают в памяти от 4 до 6. ПАСКАЛЬ допускает представление вещественных значений в виде как с плавающей, так и с фиксированной точкой.

Таблица 11

Вещественные типы данных

Тип	Диапазон значений	Мантисса	Требуемая память
real	2.9∗10E-39.. 1.7∗10E38	11 – 12	6 байт
single	1.5∗10E-45.. 3.4∗10E38	7 – 8	4 байта
double	5.0∗10E-324.. 1.7∗10E308	15 – 16	8 байт
extended	1.9 ∗10E-4951.. 1.1∗10E4932	19 – 20	10 байт
comp	-2E+63+1 .. 2E+63-1	10 – 20	8 байт

К вещественным типам данных применяются арифметические операции операции (кроме div и mod), операции отношения и стандартные функции (кроме Succ(X) и Pred(X)), что и для целых типов данных.

Логический тип (Boolean) определяет те данные, которые могут принимать логические значения TRUE и FALSE.

Для данных логического типа определены следующие логические операции

NOT - отрицание: NOT(True)=false;

OR - логическое сложение(ИЛИ): True OR False = True;

AND - логическое умножение(И): False AND True = False;

XOR - исключающее ИЛИ: True XOR True = False;

True XOR False = True;

Логический тип определен таким образом, что FALSE < TRUE. Это позволяет применять к булевским операндам все операции отношения.

В ТУРБО ПАСКАЛЬ введены еще разновидности логического типа: ByteBool, WordBool и LongBool, которые занимают в памяти ЭВМ один, два и четыре байта соответственно.

Рассмотрим пример объявления логических данных в разделе объявлений программы:

```
const
a=true;
var
b:boolean;
```

Символьный тип (Char) определяет упорядоченную совокупность символов, допустимых в данной ЭВМ. Значение символьной переменной или константы - это один символ из допустимого набора.

Символьная константа может записываться в тексте программы тремя способами:

-как один символ, заключенный в апострофы, например: 'A' 'a' 'Ю' 'ю';

-с помощью конструкции вида #K, где K - код соответствующего символа, при этом значение K должно находиться в пределах 0..255;

-с помощью конструкции вида ^C, где C - код соответствующего управляющего символа, при этом значение C должно быть на 64 больше кода управляющего символа.

К величинам символьного типа применимы все операции отношения. Для величин символьного типа определены две функции преобразования Ord(C) и Chr(K).

Первая функция определяет порядковый номер символа C в наборе символов, вторая определяет по порядковому номеру K символ, стоящий на K-ом месте в наборе символов. Порядковый номер имеет целый тип.

К аргументам символьного типа применяются функции, которые определяют предыдущий и последующий символы:

Pred(C) Succ(C). Pred('F') = 'E' ; Succ('Y') = 'Z' .

При отсутствии предыдущего или последующего символов значение соответствующих функций не определено.

Для литер из интервала 'a'..'z' применима функция UpCase(C), которая переводит эти литеры в верхний регистр 'A'..'Z'.

Рассмотрим пример объявления переменной символьного типа:

```
var
cimv:char;
const
vsym='A';
```

Адресный тип (Pointer) определяет переменные, которые могут содержать значения адресов данных или фрагментов программы. Для хранения адреса требуются два слова (4 байта), одно из них определяет сегмент, второе - смещение.

Данные *строкового типа* представляют собой последовательности символов переменной длины (от 1 до 255). Такие данные можно описывать в программах следующим образом:

```
var
vystr:string[18];{vysrt - 18 символов }
str1:string[88];{str1- 88 символов}
str2:string;{str2- 255 символов по умолчанию}
const
one_str='PASKAL';
two_str='#13#70';
```

Фактическую длину строки SS можно определить с помощью стандартной функции length(SS).

К любому символу строки можно обратиться по его номеру, например: str1[2],str2[200].

К строкам можно применять операцию сцепления или конкатенации "+": vystr + one_str, 'Моя '+'программа',

а также операции отношения , которые выполняются над строками посимвольно слева направо с учетом кодов сравниваемых символов.

Рассмотрим некоторые функции определенные над символьными данными:

copy(st,index,count)- функция копирует из строки st count символов, начиная с символа с номером index.

pos(subst,st)- функция отыскивает в строке st первое вхождение подстроки subst.>

Перечисляемый тип представляет собой ограниченную упорядоченную последовательность скалярных констант, составляющих данный тип. Значение каждой константы задается ее именем. Имена отдельных констант отделяются друг от друга запятыми, а вся совокупность констант, составляющих данный перечисляемый тип, заключается в круглые скобки.

Программист объединяет в одну группу в соответствии с каким - либо признаком всю совокупность значений, составляющих перечисляемый тип. Например, перечисляемый тип Rainbow(РАДУГА) объединяет скалярные значения:

RED, ORANGE, YELLOW, GREEN, LIGHT_BLUE, BLUE, VIOLET (КРАСНЫЙ, ОРАНЖЕВЫЙ, ЖЕЛТЫЙ, ЗЕЛЕНЫЙ, ГОЛУБОЙ, СИНИЙ, ФИОЛЕТОВЫЙ).

Перечисляемый тип Traffic_Light (СВЕТОФОР) объединяет скалярные значения RED, YELLOW, GREEN (КРАСНЫЙ, ЖЕЛТЫЙ, ЗЕЛЕНЫЙ).

Перечисляемый тип описывается в разделе описания типов, который начинается со служебного слова type, например:

type

Rainbow = (RED, ORANGE, YELLOW, GREEN, LIGHT_BLUE, BLUE, VIOLET);

Каждое значение является константой своего типа и может принадлежать только одному из перечисляемых типов, заданных в программе. Напри-

мер, перечисляемый тип Traffic_Light не может быть определен в одной программе с типом Rainbow, так как оба типа содержат одинаковые константы.

Описание переменных, принадлежащих к скалярным типам, которые объявлены в разделе описания типов, производится с помощью имен типов. Например:

type Traffic_Light= (RED, YELLOW, GREEN);

var Section: Traffic_Light;

Это означает, что переменная Section может принимать значения RED, YELLOW или GREEN.

Переменные перечисляемого типа могут быть описаны в разделе описания переменных, например:

var Section: (RED, YELLOW, GREEN);

При этом имена типов отсутствуют, а переменные определяются совокупностью значений, составляющих данный перечисляемый тип.

К переменным перечисляемого типа может быть применим оператор присваивания:

Section:= YELLOW;

Упорядоченная последовательность значений, составляющих перечисляемый тип, автоматически нумеруется, начиная с нуля и далее через единицу. Отсюда следует, что к перечисляемым переменным и константам могут быть применены операции отношения и стандартные функции Pred, Succ, Ord.

Переменные и константы перечисляемого типа не могут быть элементами списка ввода или вывода.

Отрезок любого порядкового типа может быть определен как *интервальный или ограниченный тип.*

Отрезок задается диапазоном от минимального до максимального значения констант, разделенных двумя точками. В качестве констант могут быть использованы константы, принадлежащие к целому, символьному, логическому или перечисляемому типам. Скалярный тип, на котором строится отре-

зок, называется базовым типом. Минимальное и максимальное значения констант называются нижней и верхней границами отрезка, определяющего интервальный тип. Нижняя граница должна быть меньше верхней. Над переменными, относящимися к интервальному типу, могут выполняться все операции и применяться все стандартные функции, которые допустимы для соответствующего базового типа. При использовании в программах интервальных типов данных может осуществляться контроль за тем, чтобы значения переменных не выходили за границы, введенные для этих переменных в описании интервального типа.

7.2.2. Структурированные типы

Массивы представляют собой ограниченную упорядоченную совокупность однотипных величин. Каждая отдельная величина называется компонентой массива. Тип компонент может быть любым, принятым в языке ПАСКАЛЬ, кроме файлового типа. Тип компонент называется базовым типом. Вся совокупность компонент определяется одним именем. Для обозначения отдельных компонент используется конструкция, называемая переменной с индексом или с индексами:

A[5] S[k+1] B[3,5].

В качестве индекса может быть использовано выражение. Тип индексов может быть только интервальным или перечисляемым. Действительный и целый типы недопустимы. Индексы интервального типа, для которого базовым является целый тип, могут принимать отрицательные, нулевое и положительные значения.

В операторной части программы один массив может быть присвоен другому, если их типы идентичны, например:

R1:=Z.

Для ввода или вывода массива в список ввода или вывода помещается переменная с индексом, а операторы ввода или вывода выполняются в цикле. Первый индекс определяет номер строки, второй - номер столбца. Двумер-

ные массивы хранятся в памяти ЭВМ по строкам. *Инициализация массивов* (присвоение начальных значений всем компонентам массивов) осуществляется двумя способами.

Первый способ - с использованием типизированных констант, например:

```
type Dim10= Array[1..10] of Real;
const raM10: Dim10 = ( 0, 2.1, 4, 5.65, 6.1, 6.7, 7.2, 8, 8.7, 9.3 );
```

При инициализации двумерных массивов значения компонент каждого из входящих в него одномерных массивов записывается в скобках:

```
type Dim3x2= Array[1..3,1..2] of Integer;
const iaM3x2: Dim3x2= ( (1, 2) (3, 4) (5, 6) );
```

Второй способ инициализации - использование разновидности процедуры FillChar:

```
FillChar( var V; NBytes: Word; B: Byte );
```

Эта процедура заполняет участок памяти однобайтовым значением. Например, для обнуления массива A[1..10] of Real можно записать:

```
FillChar(A, 40, 0);
```

или

```
FillChar(A, SizeOf(A), 0);
```

Запись – это структура данных, состоящая из фиксированного числа компонентов, называемых полями записи. С помощью записи (record) представляется некоторая структура статических данных. Тип данных record предоставляет программисту возможность объединить в одну связную структуру различные по типу и смыслу элементы (поля). Причем *элементами записи* могут быть и структурированные типы данных, например массивы и другие (подчиненные) записи. Для обработки доступна, как вся запись целиком, так и отдельные ее поля. Под структурой данных обычно понимают данные, объединенные в упорядоченное множество. Особо удобны записи при обработке взаимосвязанных разнородных данных.

В качестве примера приведем запись, представляющую информацию из адресного справочника:

```
var reference_book = record
surname,
name,
address,
city : string[20];
post_index : string[4];
telerhone : string[12];
end;
```

При обращении к отдельным полям указывается имя всей записи и имя отдельного поля, причем они разделяются точкой. Так, с помощью операции

```
reference_book. surname :='Иванов';
```

имя записи поле

Вы присваиваете полю surname записи reference_book значение 'Иванов'. Таким способом можно обратиться к любому полю, любой записи, как для занесения туда некоторого значения, так и для извлечения необходимой информации. Следует только применять эти операции с учетом типа данных обрабатываемой записи.

Понятие *множества* в языке ПАСКАЛЬ основывается на математическом представлении о множествах: это ограниченная совокупность различных элементов. Для построения конкретного множественного типа используется перечисляемый или интервальный тип данных. Тип элементов, составляющих множество, называется базовым типом.

Множественный тип описывается с помощью служебных слов Set of, например:

```
type M= Set of B;
```

Здесь M - множественный тип, B - базовый тип.

Пример описания переменной множественного типа:

```
type
```

M= Set of 'A'..'D';

var

MS: M;

Принадлежность переменных к множественному типу может быть определена прямо в разделе описания переменных:

var C: Set of 0..7;

Константы множественного типа записываются в виде заключенной в квадратные скобки последовательности элементов или интервалов базового типа, разделенных запятыми, например:

['A', 'C'] [0, 2, 7] [3, 7, 11..14].

Константа вида [ ] означает пустое подмножество.

Множество включает в себя набор элементов базового типа, все подмножества данного множества, а также пустое подмножество. Если базовый тип, на котором строится множество, имеет К элементов, то число подмножеств, входящих в это множество, равно 2 в степени К. Пусть имеется переменная Р интервального типа:

var P: 1..3;

Эта переменная может принимать три различных значения - либо 1, либо 2, либо 3. Переменная Т множественного типа

var T: Set of 1..3;

может принимать восемь различных значений:

[ ] [1,2] [1] [1,3] [2] [2,3] [3] [1,2,3]

Порядок перечисления элементов базового типа в константах безразличен.

Значение переменной множественного типа может быть задано конструкцией вида [T], где Т - переменная базового типа.

К переменным и константам множественного типа применимы операции присваивания(:=), объединения(+), пересечения(*) и вычитания(-):

['A','B'] + ['A','D'] даст ['A','B','D']

['A'] * ['A','B','C'] даст ['A']

['A','B','C'] - ['A','B'] даст ['C'].

Результат выполнения этих операций есть величина множественного типа.

К множественным величинам применимы операции: тождественность (=), нетождественность (<>), содержится в (<=), содержит (>=). Результат выполнения этих операций имеет логический тип, например:

['A','B'] = ['A','C'] даст FALSE

['A','B'] <> ['A','C'] даст TRUE

['B'] <= ['B','C'] даст TRUE

['C','D'] >= ['A'] даст FALSE.

Кроме этих операций для работы с величинами множественного типа в языке ПАСКАЛЬ используется операция

in,

проверяющая принадлежность элемента базового типа, стоящего слева от знака операции, множеству, стоящему справа от знака операции. Результат выполнения этой операции - булевский. Операция проверки принадлежности элемента множеству часто используется вместо операций отношения, например:

A in ['A', 'B'] даст TRUE,

2 in [1, 3, 6] даст FALSE.

При использовании в программах данных множественного типа выполнение операций происходит над битовыми строками данных. Каждому значению множественного типа в памяти ЭВМ соответствует один двоичный разряд. Например, множество

['A','B','C','D'] представлено в памяти ЭВМ битовой строкой

1 1 1 1.

Подмножества этого множества представлены строками:

['A','B','D'] 1 1 0 1

['B','C'] 0 1 1 0

['D'] 0 0 0 1

Величины множественного типа не могут быть элементами списка ввода - вывода.

В каждой конкретной реализации транслятора с языка ПАСКАЛЬ количество элементов базового типа, на котором строится множество, ограничено. В TURBO PASCAL количество базовых элементов не должно превышать 256.

Инициализация величин множественного типа производится с помощью типизированных констант:

```
const seLit: Set of 'A'..'D'= [];
```

7.2.3. Константы

Тип констант в языке ПАСКАЛЬ определяется по их виду: константы целого типа - это целые числа, не содержащие десятичной точки, константы действительного типа - действительные числа, логические константы - логические значения TRUE и FALSE, символьные константы - либо строки длиной в один символ, либо конструкции вида #K или ^K.

Язык ПАСКАЛЬ допускает использовать синонимы для обозначения констант, в этом случае текст программы содержит раздел описания констант.

7.2.4. Ввод-вывод данных

Рассмотрим организацию ввода и вывода данных с терминального устройства. Терминальное устройство - это устройство, с которым работает пользователь, обычно это экран (дисплей) и клавиатура.

Для ввода и вывода данных используются стандартные процедуры ввода и вывода Read и Write, оперирующие стандартными последовательными файлами INPUT и OUTPUT.

Эти файлы разбиваются на строки переменной длины, отделяемые друг от друга признаком конца строки. Конец строки задается нажатием клавиши ENTER.

Для ввода исходных данных используются операторы процедур ввода:

Read(A1,A2,...AK);

ReadLn(A1,A2,...AK);

ReadLn;

Первый из них реализует чтение К значений исходных данных и присваивание этих значений переменным А1, А2, ..., АК. Второй оператор реализует чтение К значений исходных данных, пропуск остальных значений до начала следующей строки, присваивание считанных значений переменным А1, А2, ..., АК. Третий оператор реализует пропуск строки исходных данных.

При вводе исходных данных происходит преобразование из внешней формы представления во внутреннюю, определяемую типом переменных. Переменные, образующие список ввода, могут принадлежать либо к целому, либо к действительному, либо к символьному типам. Чтение исходных данных логического типа в языке ПАСКАЛЬ недопустимо.

Операторы ввода при чтении значений переменных целого и действительного типа пропускает пробелы, предшествующие числу. В то же время эти операторы не пропускают пробелов, предшествующих значениям символьных переменных, так как пробелы являются равноправными символами строк. Пример записи операторов ввода:

```
var rV, rS: Real;
iW, iJ: Integer;
chC, chD: Char;
.....
Read(rV, rS, iW, iJ);
Read(chC, chD);
```

Значения исходных данных могут отделяться друг от друга пробелами и нажатием клавиш табуляции и Enter.

Для вывода результатов работы программы на экран используются операторы:

Write(A1,A2,...AK);

WriteLn(A1,A2,...AK);

WriteLn;

Первый из этих операторов реализует вывод значений переменных A1, A2,...,AK в строку экрана. Второй оператор реализует вывод значений переменных A1, A2, ..., AK и переход к началу следующей строки. Третий оператор реализует пропуск строки и переход к началу следующей строки.

Переменные, составляющие список вывода, могут относиться к целому, действительному, символьному или булевскому типам. В качестве элемента списка вывода кроме имен переменных могут использоваться выражения и строки.

Вывод каждого значения в строку экрана происходит в соответствии с шириной поля вывода, определяемой конкретной реализацией языка. Форма представления значений в поле вывода соответствует типу переменных и выражений: величины целого типа выводятся как целые десятичные числа, действительного типа - как действительные десятичные числа с десятичным порядком, символьного типа и строки - в виде символов, логического типа - в виде логических констант TRUE и FALSE.

Оператор вывода позволяет задать ширину поля вывода для каждого элемента списка вывода. В этом случае элемент списка вывода имеет вид A:K, где A - выражение или строка, K - выражение либо константа целого типа. Если выводимое значение занимает в поле вывода меньше позиций, чем K, то перед этим значением располагаются пробелы. Если выводимое значение не помещается в ширину поля K, то для этого значения будет отведено необходимое количество позиций. Для величин действительного типа элемент списка вывода может иметь вид A:K:M, где A - переменная или выражение действительного типа, K - ширина поля вывода, M - число цифр дробной части выводимого значения. K и M - выражения или константы целого типа. В этом случае действительные значения выводятся в форме десятичного числа с фиксированной точкой.

Пример записи операторов вывода:

. . . . . .

```
var rA, rB: Real; iP,iQ:Integer;
bR, bS: Boolean; chT, chV, chU, chW: Char;
. . . . . .
WriteLn(rA, rB:10:2);
WriteLn(iP, iQ:8);
WriteLn(bR, bS:8);
WriteLn(chT, chV, chU, chW);
```

7.3. Операторы

7.3.1. Условный оператор

Условный оператор позволяет проверить некоторое условие и в зависимости от результатов проверки выполнить то или иное действие. Таким образом ,условный оператор-это средство ветвления вычислительного процесса.

Структура условного оператора:

if<условие> **then** <оператор1> **else** <оператор2>.

if, then, else – зарезервированные слова (если, то, иначе);

<условие> – произвольное выражение логического типа;

<оператор1>,<оператор2> – любые операторы языка Турбо Паскаль.

Алгоритм работы условного оператора

Условный оператор работает по следующему алгоритму. Вначале вычисляется условное выражение <условие>. Если результат true (истина),то выполняется <оператор1>, а <оператор2> пропускается. Если результат есть false (ложь), наоборот, <оператор1> пропускается, и выполняется <оператор2>.

Пример:(фрагмент программы)

```
Var
x,y,max:real;
```

.....

if x>max **then** y:=max **else** y:=x;

При выполнении этого фрагмента программы, переменная y получит значение x, если только это значение не превышает max, в противном случае y станет равным max.

В качестве условия могут использоваться различные условные выражения :

x>0,

(x>9) **and** (y<2)

(f<>r) **or** (r>=4)

not (ttt<=1)

Часть else <оператор 2> условного оператора может быть опущена. Тогда при значении true условного выражения выполняется <оператор1>, в противном случае этот оператор пропускается.

Пример:(фрагмент программы)

Var

x,y,max:real;

.....

if x>max **then**

begin

max:=x;

write(max);

end;

y:=x;

В этом примере переменная y всегда будет иметь значение переменной x, а в max запоминать максимальное значение x.

После **THEN/ELSE** может располагаться либо один оператор, либо последовательность операторов, заключенная в операторные скобки **BEGIN END**. Такая последовательность операторов называется *составным оператором.*

Поскольку любой из операторов <оператор1> и <оператор> может быть любого типа, в том числе и условным, а в то же время не каждый из <<вложенных>> условных операторов может иметь часть **else** <оператор2>,то возникает неоднозначность трактовки условий. Эта встречающаяся часть **else** соответствует ближайшей к ней сверху части **then** условного оператора.

Зарезервированному слову **else** в операторе **if** не должна предшествовать точка с запятой. Следует помнить, что условный оператор управляет только одним оператором . Для обеспеченности однозначности в языке Паскаль принято соглашение о том, что каждому **else** соответствует предыдущий свободный **if**.

if a>b **then**

if c<0 **then**

write(c)

else

c:=0;

7.3.2. Операторы цикла с условиями

В языке ТУРБО ПАСКАЛЬ существуют два оператора цикла с условиями. С их помощью можно запрограммировать повторяющиеся фрагменты программы. Операторы цикла с условиями проверяют условия выполнения или повторения цикла. Рассмотрим их.

Оператор цикла с предусловием

WHILE <условие> **DO** <оператор>;

здесь **WHILE, DO** – зарезервированные слова (пока [выполняется условие], делать);

<условие> – выражение логического типа;

<оператор> – произвольный оператор .

Логическое выражение <условие> определяет будет ли цикл выполняться или завершит свою работу. Если выражение <условие> имеет значение TRUE (истина), то выполняется <оператор>, после чего вычисление выражения <условие> и его проверка повторяются. Если <условие> имеет значение FALSE (ложь), оператор **WHILE** прекращает свою работу, т.е. цикл заканчивается.

В качестве <оператора> может быть либо оператор , либо составной оператор, т.е. последовательность операторов, заключенная в операторные скобки **BEGIN END**.

Оператор цикла с постусловием

REPEAT <тело цикла> **UNTIL** <условие>;

здесь **REPEAT, UNTIL –** зарезервированные слова (повторять до тех пор,[пока не будет выполнено условие]);

<тело цикла> – произвольная последовательность операторов;

<условие> – выражение логического типа.

Операторы <тело цикла> выполняются хотя бы один раз, после чего вычисляется выражение <условие>: если его значение есть FALSE (ложь), операторы <тело цикла> повторяются, в противном случае оператор **REPEAT...UNTIL** завершает свою работу.

Отличия двух операторов цикла с условиями.

-В операторе цикла с предусловием возможна такая ситуация, что операторы, содержащиеся внутри него не будут выполнены ни одного раза, т.к. условие выхода из цикла там проверяется перед выполнением оператора. В операторе с постусловием оператор, содержащийся в цикле выполняется хотя бы один раз.

-В этих двух операторах по-разному проверяется логическое выражение на выход из цикла.

Выход из цикла.

В циклах с условиями при большом объеме вычислений возможна ситуация, когда время выполнения цикла становится слишком большим. В этом случае необходимо использовать стандартную процедуру прекращения выполнения цикла **BREAK**.

Эта процедура используется т.ж. для досрочного выхода из цикла в случае получения результата до конца выполнения цикла.

Процедура **CONTINUE** используется для досрочного перехода к следующей операции цикла.

7.3.3. Оператор цикла FOR

Структура оператора:

FOR <пар_цик>:=<нач_знач> **TO** <кон_знач> **DO** <оператор>.

FOR, TO, DO – зарезервированные слова (для, до, выполнить);

<пар_цик> – параметр цикла;

<нач_знач> – начальное значение;

<кон_знач> – конечное значение;

<оператор> – произвольный оператор.

При выполнении оператора **FOR** вначале вычисляется выражение <нач_знач> и осуществляется присваивание <пар_цик>:=<нач_знач>. После этого циклически повторяется:

-проверка условия <пар_цик><=<кон_знач>; если условие не выполнено, оператор **FOR** завершает свою работу;

-выполнение оператора <оператор>;

-наращивание переменной <пар_цик> на единицу.

Другая форма оператора FOR:

FOR <пар_цик>:=<нач_знач> **DOWNTO** <кон_знач> **DO** <оператор>

7.4. Операции с одномерными массивами

При программировании целого ряда задач единственно рациональным подходом является использование структур данных, именуемых массивами. Массивы являются, пожалуй, наиболее важным структурным типом в любом языке программирования.

Многие алгоритмы обработки данных, такие как вычисление общих характеристик большого числа данных, поиск наибольшего (наименьшего) значения, сортировка последовательностей при использовании массивов становятся нагляднее и понятнее для учащихся.

Под структурой данных типа *массив* понимают однородную структуру однотипных данных, одновременно хранящихся в последовательных ячейках оперативной памяти. Эта структура должна иметь имя и определять заданное количество данных (элементов). Однотипность данных определяет возможность использования циклических алгоритмов для обработки всех элементов массива. Количество итераций цикла определяется количеством элементов массива. Одновременное хранение в памяти всех элементов массива позволяет решать большой набор задач, таких как, поиск элементов, упорядочение и изменение порядка следования элементов.

Доступ к любому элементу массива осуществляется по его номеру (индексу). Поэтому для обращения к элементу массива используют имя_массива(номер элемента), например, A(5).

Если за каждым элементом массива закреплен только один его порядковый номер, то такой массив называется *линейным.* Вообще количество индексов элементов массива определяет *размерность массива*. По этом признаку массивы делятся на *одномерные (линейные), двумерные, трёхмерные* и т.д.

В информатике и математике массив называется *одномерным*, если для получения доступа к его элементам достаточно одной индексной переменной. Так, чтобы найти в шкафу с одним рядом ящиков нужный Вам ящик, достаточно знать его номер и точку начала отсчета.

В ПАСКАЛЕ 7.0 Вы можете объявить одномерный массив следующим образом:

```
VAR
имя массива:ARRAY[нач индекс..кон индекс] OF тип данных;
```

ПРИМЕРЫ: var a:array[1..5] of byte;

```
b:array[1..3 of char;
z:array['d'..'g'] of integer;
```

Величины, соответствующие начальному и конечному индексам, т.е. значения, указанные в квадратных скобках, разделяются двумя точками.

В TP можно определить массив любой размерности. Ограничения, которые имеются при описании массивов:

1)границы изменения индексов необходимо открыть либо в самом описании массива, либо до его описания;

2)максимальный объем памяти выделенной под массив не должен превышать 64кБ.

7.4.1. Ввод элементов массива

Ввод элементов массива с клавиатуры:

```
FOR I:=1 TO n DO
READ(A[I]);
```

Ввод элементов массива с помощью типизированных констант:

```
CONST
A1:ARRAY[1..6] of integer =(-5,8,5,0,7,-8);
A2:ARRAY[1..3] of char=('a','b','c');.
```

Ввод элементов массива с помощью датчика случайных чисел:

```
FOR i:=1 TO 10 DO
BEGIN A[i]:=random (20);
WRITE(A[i]:5);
END;
```

В этом случае значениями элементов массива A[i] будут произвольные значения от 0 до 19.

Для того чтобы получились дробные числа нужно в функции random опустить параметр.

7.4.2. Сортировка массива

Под *сортировкой* понимают процесс перестановки объектов данного массива в определенном порядке. Целью сортировки являются упорядочение массивов для облегчения последующего поиска элементов в данном массиве. Рассмотрим основные алгоритмы сортировки по возрастанию числовых значений элементов массивов. Существует много методов сортировки массивов. В этой работе будут рассмотрены алгоритмы двух методов: модифицированного метода простого выбора и метода парных перестановок.

Сортировка модифицированным методом простого выбора

Этот метод основывается на алгоритме поиска минимального элемента. В массиве A(1..n) отыскивается минимальный элемент, который ставится на первое место . Для того, чтобы не потерять элемент , стоящий на первом месте , этот элемент устанавливается на место минимального . Затем в усеченной последовательности, исключая первый элемент, отыскивается минимальный элемент и ставится на второе место и так далее n-1 раз пока не встанет на свое место предпоследний n-1 элемент массива A, сдвинув максимальный элемент в самый конец.

Сортировка методом парных перестановок

Самый простой вариант этого метода сортировки массива основан на принципе сравнения и обмена пары соседних элементов.

Процесс перестановок пар повторяется просмотром массива с начала до тех пор , пока не будут отсортированы все элементы , т.е. во время очередного просмотра не произойдет ни одной перестановки. Для подсчета количества перестановок целесообразно использовать счетчик - специальную переменную B. Если при просмотре элементов массива значение счетчика пере-

становок осталось равным нулю, то это означает, что все элементы отсортированы

Пример

Вычислить сумму элементов массива

```
program a8;
var
a:array[1..10]of real;
i:integer;
c:real;
begin
c:=0;
writeln('Вычислить сумму элементов массива');
writeln(' Введите 10 элементов массива');
for i:=1 to 10 do
begin
write(i,' элемент= ');
readln(a[i]);
c:=c+a[i];
end;
writeln('Сумма элементов массива = ',c);
writeln('Для окончания работы нажмите enter');
readln;
readln;
end.
```

Пример

В одномерном массиве найти сумму положительных, произведение отрицательных, количество четных элементов

```
program a6;
var
a:array[1..10] of integer;
```

```
e,i,c,d:integer;
begin
writeln(' В одномерном массиве найти сумму положительных,');
writeln(' произведение отрицательных , кол-во четных элементов');
writeln;
writeln('Введите массив из 10 элементов через пробелы')
d:=0;
c:=1;
e:=0;
for i:=1 to 10 do
begin
read (a[i]);
if a[i]>0 then d:=d+a[i] ;
if a[i]<0 then c:=c*a[i] ;
if a[i] mod 2 =0 then e:=e+1 ;
end;
writeln;
writeln('Результат:');
if d<>0 then writeln('Сумма положительных элементов d= ',d)
else writeln('Положительных нет.');
if c<>1 then writeln('Произведение отрицательных элементов c= ',c)
else writeln('Отрицательных нет.');
if e<>0 then writeln('Кол-во четных элементов e= ',e)
else writeln('Четных нет.');
readln;
readln;
end.
```

7.5. Алгоритмы обработки двумерных массивов.

Двумерный массив – это структура однотипных элементов, расположенных в виде таблицы значений.

1 4 1 0

4 9 1 3

8 5 0 1

1 2 3 4

Рисунок 24 – Пример двумерного массива

Такое представление значений соответствует математическому понятию двумерный массив. Каждый элемент в двумерном массиве идентифицируется *номером строки и номером столбца*, на пересечении которых он расположен. Например, в двумерном массиве A, изображенном на рис. 24, элемент со значением 5 расположен на пересечении третьей строки и второго столбца. Этот элемент будет обозначаться как A(3,2). А элемент A(1,4) имеет значение, равное нулю. Такое представление набора значений позволяет выполнять обработку как отдельных значений в двумерном массиве, так и последовательности значений, расположенных в строках или столбцах. В дальнейшем будем считать, что для двумерного массива A(N,M) в обозначении элемента A(i,j) первое значение i соответствует номеру строки и изменяется от1 до N, а j - номеру столбца и изменяется от 1 до M. . В отличие от одномерного массива, в котором использовался только один номер для определения местоположения элемента и требовался только один цикл для ввода элементов, в двумерном массиве для обработки элементов необходимы два вложенных друг в друга цикла. Внешний цикл предназначен для изменения номера строки i, а второй, внутренний, - для изменения номера столбца j в текущей строке i.

7.5.1.Описание элементов массива

С помощью типизированной переменной:

```
type
mar=array[1..5] of byte;
var
a:array[1..6] of mar;
```

В разделе описания переменных:

```
m:array[1..5] of array [1..8] of string;
a:array[1..5,1..4] of integer;
```

В Паскале можно определить (описать) массив любой размерности. Ограничения, которые имеются при описании массивов следующие:

1) границы изменения индексов (размерность) необходимо определить либо в описании массива, либо до его описания.

```
const
N=5; M=4;
var
a:array [1..N,1..M] of real;
```

2) максимальный объем памяти, выделяемый под массив не должен превышать 64 КБайта.

7.5.2. Ввод элементов массива

Ввод элементов массива с клавиатуры.

```
for i:=1 to n do
for j:=1 to m do
read(a[i,j]);
```

Ввод элементов массива с помощью типизированной константы:

```
const
c:array[1..3,1..5] of byte = ((0,1,2,3,4),
(5,6,7,8,9),
(2,1,0,3,4));
```

Ввод элементов массива с помощью датчика случайных чисел:

```
randomize; { запуск генератора случайныхчисел}
for i:=1 to 5 do
for j:=1 to 7 do
a[i,j]:=random(20);
```

В этом случае элементами массива будут числа от 0 до 19.

Ввод элементов массива из файла:

```
var f:text;
A:array[1..5,1..7] of integer;
begin
assign(f,'srsdf.txt');
reset(f);
for i:=1 to 5 do
for j:=1 to 7 do
read(f,a[i,j]);
close(f);
end;
```

7.5.3. Вывод элементов массива

Вывод элементов массива в строку:

```
for i:=1 to 5 do
for j:=1 to 7 do
write(a[i,j]);
```

Вывод элементов массива в матричной форме:

```
for i:=1 to 5 do
begin
for j:=1 to 7 do
write(a[i,j]:3);
writeln;
end;
```

Вывод элементов массива в файл:

```
var f:text;
a:array[1..5,1..7] of integer;
bagin
```

```
assign(f,'sdfg.txt');
rewrite(f);
for i:=1 to 5 do
for j:=1 to 7 do
write(f,a[i,j]);
close(f);
```

7.5.4. Работа с элементами двумерного массива

В ниже приведенных задачах примем условные обозначения:

<read> —ввод элементов массива одним из выше приведенных способов;

<write> —вывод элементов массива одним из выше приведенных способов.

1. Задачи, в которых требуется выполнить поэлементную обработку некоторых компонент массива, удовлетворяющих определенному условию.

a) Подсчитать сумму положительных элементов в двумерном массиве.

```
var
a:array[1..5,1..7] of integer;
i,j:byte;
k:integer;
begin
<read>;
k:=0;
for i:=1 to 5 do
for j:=1 to 7 do if a[i,j]>0 then k:=k+a[i,j];
writeln(k);
end.
```

b) Определить произведение элементов в каждой строке.

```
var
a:array[1..4,1..4] of real;
```

```
i,j:byte;
p:real;
begin
<read>;
p:=1;
for i:=1 to 4 do begin
for j:=1 to 4 do
p:=p*a[i,j];
write(p);
end;
end.
```

c) В двумерном массиве найти сумму максимального и минимального элементов, если она меньше нуля, тогда все отрицательные элементы матрицы, лежащие выше главной диагонали, заменить на нули.

```
Program two;
Var a:array[1..4,1..4] of integer;
I, j, p, min,max: integer;
Begin
<read>;
max:=a[1,1];
for I:=1 to 4 do {поиск максимального элемента}
for j:=1 to 4 do if a[I, j]>max then max:=a[I, j];
min:=a[1,1];
for I:=1 to 4 do {поиск минимального элемента}
for j:=1 to 4 do if a[I, j]<min then min:=a[I, j];
p:=min+max;
if p<0 then begin
for I:=1 to 4 do {замена элементов по условию}
for j:=1 to 4 do if (I<j) and (a[I, j]<0) then a[I, j]:=0;
<writeln>; {вывод измененного массива}
```

End.

2. Задачи, в которых требуется в качестве обработки изменить порядок следования элементов в исходном массиве.

а) Поменять местами два элемента в двумерном массиве.

```
Program four;
Var a:array[1..4,1..4] of integer;
I, j, f, e, g, s, k:integer;
b:boolean;
begin
<read>;
repeat
b:=false;
writeln(‘введите номера строк’);
readln(f);
readln(e);
writeln(‘введите номера столбцов’);
readln(g);
readln(s);
if (f<=5) and (e<=5) and (g<=5) and (s<=5) then b:=true
else
writeln(‘Массив 5*5. Введите числа<5’);
until b=true;
k:=a[e,s];
a[e,s]:=a[f,g]; {перестановка элементов}
a[f,g]:=k;
<write>; {вывод измененного массива}
end.
```

7.6. Подпрограммы

7.6.1. Основные понятия

В системе программирования Турбо Паскаль концепция подпрограмм реализуется в виде *подпрограмм-процедур* и *подпрограмм-функций*. Подпрограммы, реализованные в системе Турбо Паскаль, называются стандартными, с некоторыми вы уже познакомились.

В качестве примера приведем стандартные функции sin(x), cos(x), length(s), eof(f) и стандартные процедуры READ(x,y), WRITE(x,y).

Основное отличие функций от процедур состоит в том, что они предназначены для вычислений единственного значения и могут использоваться в выражениях, например: X := Sin(A);

В процедурах можно вычислять сколь угодно много значений или вообще ни одного.

Кроме стандартных подпрограмм Вы можете объявить и использовать свои подпрограммы (пользовательские), реализующие отдельные алгоритмы и вычисления. Отличие стандартных подпрограмм от пользовательских заключается в том, что пользовательские подпрограммы требуется описать в разделе описания программы, а затем уже использовать.

Подпрограммы являются одним из основных средств структурирования программ, то есть представления программы в виде отдельных синтаксических конструкций, имеющих уникальное имя, и предназначенных для выполнения определенного алгоритма.

Применение подпрограмм дает возможность уменьшить число повторений одной и той же последовательности операторов, а также конструировать программу как набор отдельных подпрограмм. Это позволяет получить более логичный процесс программирования. Каждая процедура или функция определяется только один раз, но может использоваться многократно.

Структура процедур и функций аналогична структуре полной программы на языке Турбо Паскаль. В процедурах и функциях могут быть описаны собственные метки, константы, типы, собственные переменные и даже соб-

ственные процедуры и функции. Внутренние описания должны следовать в том же порядке, что и разделы основной программы.

Структура программы, использующей подпрограммы, является многоуровневой, то есть внутри каждой подпрограммы может содержаться любое количество других подпрограмм. Все объекты, описанные на уровне главной программы, называются глобальными. Они доступны для использования внутри любой подпрограммы, входящей в состав основной программы. Объекты, описанные в виде локальных переменных, располагаются внутри соответствующей подпрограммы. Если в некоторой процедуре описан объект, имя которого совпадает с именем глобального объекта, то в этом случае этот глобальный объект становится недоступным в данной подпрограмме.

В Паскале программы имеют строгий формат:

```
program     Имя-программы
label       метки;
const       описание констант;
 type       определения типов данных;
var         описания переменных;
procedures и functions;
begin
основное тело программы
end.
```

Турбо-Паскаль обеспечивает значительно более гибкую структуру программы. Все, что требуется, - это, чтобы оператор program (если таковой имеется) был первым, а основное тело программы - последним. Между ними можно иметь сколь угодно много секций описания, в любом порядке и как угодно смешанными с процедурами и функциями. Но прежде, чем что-либо использовать, это должно быть определено, иначе на этапе компиляции появится сообщение об ошибке.

7.6.2. Процедуры.

Процедурами в Турбо Паскале называют вид подпрограмм. Они предназначены для оформления любой самостоятельной части программы в виде отдельной синтаксической конструкции, которая решает определенную задачу. Процедуры используются для изменения окружающей среды программы и, как правило, определяют некоторые действия.

Описание каждой процедуры начинается с заголовка, в котором задаются имя процедуры и список формальных параметров с указанием их типов. Процедура может быть и без параметров, тогда в заголовке указывается только ее имя. С помощью параметров осуществляется передача исходных данных в процедуру, а также передача результатов работы обратно в вызвавшую ее программу.

Структура процедуры:

```
Procedure <имя>(список параметров);
<раздел описания>
begin
<раздел операторов>
end;
```

Список параметров, указанный в заголовке любой подпрограммы, в том числе и процедуры, содержит имена параметров и их типы. Это формальные параметры. Заголовок процедуры может не содержать списка параметров.

Пример

```
program nat4;
{ процедура без параметров }
procedure text;
begin
writeln(' Я обучаюсь на естественнонаучном факультете ');
writeln(' Ульяновского Государственного Технического ');
writeln(' Университета по специальности прикладная ');
writeln(' математика и информатика. ');
```

```
end;{text}
{***MAIN***}
begin
writeln(' Пример 1.1: вывести текст на экран ');
{ вызов процедуры }
text;
end.
```

7.6.3. Функции

Подпрограмма-функция предназначена для вычисления какого-либо одного значения. Описание каждой функции начинается с заголовка, в котором задаются имя функции , список формальных параметров с указанием их типов и тип значения функции.

С помощью параметров осуществляется передача исходных данных в подпрограмму. Описание функции в общем случае выглядит следующим образом:

```
FUNCTION <имя>(список параметров):<тип>;
<раздел описания>
begin
<раздел операторов>
end;
```

Функция может возвратить параметры следующих типов: целого, символьного, вещественного, строкового и логического.

В разделе операторов функции хотя бы раз имени функции должно быть присвоено значение.

Пример

Пример функции вычисления факториала числа N.

```
function Factorial(n :Integer): Longint;{заголовок функции}
VAR Fact: Longint; {раздел объявлений}
```

```
i: Byte;
begin {раздел операторов}
Fact := n;
for i := n-1 downto 2 do
Fact := Fact*i;
Factorial := Fact; { вычисленное значение факториала }
end;
```

Для вызова функции из основной программы или другой подпрограммы следует в выражении, где необходимо использовать значение функции, указать имя функции со списком фактических параметров, которые должны совпадать по количеству и типам с формальными параметрами функции,

например:

```
Part := Sqr(T)/Factorial(i);
```

В этом операторе:

Sqr(T) - вызов стандартной функции возведения в квадрат с фактическим параметром T;

Factorial(i) - вызов функции, вычисляющей факториал с фактическим параметром i.

7.6.4. Формальные и фактические параметры

Формальные параметры указываются в заголовке подпрограммы, *фактические параметры* – при обращении к подпрограмме.

Само описание процедуры не вызывает конкретных действий. Для того, чтобы выполнить действия, указанные в процедуре, необходимо в вашей программе использовать оператор вызова соответствующей процедуры. Например, для вызова процедуры в нужном месте программы указывается имя этой процедуры, за которым в круглых скобках могут следовать параметры, передаваемые из программы в процедуру. Такие параметры называются *фактическими.* При их указании в этом списке типы не используются:

```
< имя >(x,y);
```

Пример: write(x,y);

При вызове процедуры из основной программы формальные параметры заменяются на фактические. Поэтому количество формальных и фактических параметров и их типы должны совпадать.

Все *формальные параметры* можно разбить на четыре категории:

1)*параметры-значения* (они в основной программе подпрограммой не меняются);

2) *параметры-переменные* (их подпрограмма может изменить в основной программе);

3)*параметры-константы* (используются только в версии 7.0);

4)*параметры без типа.*

Для каждого формального параметра следует указать имя и, как правило, тип. Имена параметров могут быть любыми, в том числе и совпадать с именами объектов программы. Необходимо лишь помнить, что в этом случае параметр основной программы с таким именем становится недоступным для непосредственного использования подпрограммой. Тип формального параметра может быть практически любым, однако в заголовке подпрограммы нельзя использовать структурный тип, например, описание массива или записи.

Например, нельзя писать:

function Max(A: array[1..100] of Real): Real;

Чтобы правильно записать этот заголовок, следует в основной программе ввести тип-массив, а затем использовать его в заголовке:

type tArr = array[1..100] of Real;

function Max(A: tArr): Real;

Пример

Функция вычисления максимального элемента массива, размер массива и его элементы определяются датчиком случайных чисел

Program Maxelement;

```
Type
tArr=array[1..100] of Integer;
Var
Massiv : tArr;
Maxim : Integer;
K ,X : Byte;
Function Max(Mas : tArr; N: Byte): Integer;
Var Ma : Integer;
i : Byte;
Begin
Ma := Mas[1];
for i := 2 to N do
if Ma < Mas[i] then
Ma := Mas[i];
Max := Ma;
End;{Max}
{**************************************}
Begin
randomize;
k := random(10) + 1; {размер массива}
write(' элементы массива :');
for x := 1 to K do
begin
Massiv[x] := random(100);
write(' ',massiv[x]);
end;
Maxim := Max(Massiv,K);
writeln(' максимальный элемент массива : ',Maxim:3);
End.
```

Другим способом передачи в подпрограмму параметра-массива является передача *массива открытого типа* без указания диапазона изменения индексов массива.

Тема 8. Технология программирования – 2 часа

Традиционная технология программирования формировалась на заре вычислительной техники, когда в распоряжении пользователей были ограниченные ресурсы ЭВМ, а разработчик программ был в то же время и главным ее пользователем. В этих условиях главное внимание обращалось на получение эффективных программ в смысле оптимального использования ресурсов ЭВМ.

В настоящее время, когда сфера применения ЭВМ чрезвычайно расширилась, разработка и эксплуатация программ осуществляется, как правило, разными людьми. Поэтому наряду с эффективностью на первый план выдвигаются и другие важные характеристики программ такие, как понятность, хорошая документированность, надежность, гибкость, удобство сопровождения и т.п. Проблема разработки программ, обладающих такими качествами, объясняется трудоемкостью процесса программирования и связанным с этим быстрым ростом стоимости программного обеспечения.

Для создания "хорошей" программы появляется необходимость придерживаться определенных принципов или определенной дисциплины программирования. Значительный прогресс в области программирования достигается с использованием так называемых структурного, модульного и объектно-ориентированного программирования.

8.1. Жизненный цикл программ

Периодически повторяющийся процесс разработки, внедрения и эволюции программного продукта принято называть его *жизненным циклом*. Современный рынок программного обеспечения требует, чтобы выпуск программ был быстрым, а его дальнейшая эксплуатация - долговременной и на-

дежной. Для достижения этого необходима хорошая организация и тщательное планирование всего жизненного цикла программного продукта. Жизненный цикл программ состоит из нескольких *этапов***.** Современные программные системы относятся к классу больших и сложных систем. Один разработчик не в состоянии охватить все аспекты ее функционирования. Для решения этой проблемы целесообразно использовать следующие стратегические подходы:

-анализ сложности задачи

-абстракция (вместо сложного реального объекта рассматривается обобщенная модель)

-декомпозиция – разделение большой системы на меньшие подсистемы , которые вновь разделяются на меньшие составляющие.

-построение иерархии выделенных компонент.

8.2. Этапы разработки программ

Программы небольшого и среднего размера (несколько тысяч строк) создаются, как правила в несколько этапов. Сначала необходимо точно *определить требования*, предъявляемые заказчиком к программному продукту. Этот этап очень важен, так как от правильного понимания целей и требований зависит в основном успех реализации.

В дальнейшем проводится *анализ предметной области* и составляется с помощью экспертов в этой области ее *формальная модель*. При этом целесообразно использовать средства автоматизации построения моделей (CASE-средств). На основе результатов моделирования составляется и утверждается *техническое задание, календарный план работ*, где указываются все сроки, конкретные исполнители и выполняемые работы.

На следующем этапе выбирается *методология разработки ПО*, технология программирования, система программирования и выполняются работы по созданию программного продукта. Первая версия разработанной програм-

мы обязательно должна быть подвергнута *тестированию* как со стороны программистов, так и со стороны заказчика на предмет выявления ошибок.

После устранения всех обнаруженных ошибок начинается *этап внедрения*. Некоторые программисты недооценивают важность этого этапа. Но на этот этап может потребоваться даже больше времени, чем на разработку программы. На этом этапе возможно придется настроить локальную сеть, инсталлировать вспомогательные программы, перенести и выверить в программу большой объем важной и оперативной информации. Этот этап обычно заканчивается подписанием акта приемки работ по созданию программы.

Немаловажным этапом является *этап обучения сотрудников* организации заказчика приемам работы с внедренной программой.

Последним этапом является *этап сопровождения программы*. На этом этапе создатели программы гарантируют на определенный период бесплатное разрешение вопросов, возникающих при эксплуатации программы, а также информируют заказчика о новых усовершенствованных версиях программы.

8.3. Качество программных продуктов

Современные технологии надежного ПО предусматривают непрерывный сквозной контроль качества разрабатываемой программы на всех этапах разработки программы. Наибольшие проблемы, связанные с качеством программ возникают из-за ошибок менеджеров в управлении процессом разработки программ. Поэтому в последнее время все большее внимание при разработке программ наряду с технологией разработки в организациях уделяется вопросам эффективного управления или менеджмента. Применяя *принципы тотального управления качеством (TQM)*, можно повысить качество программ, снизить сроки их создания.

Идеи тотального (всеобщего) менеджмента качества (Total Quality Management - TQM) возникли в начале 1980-х применительно к произ-

водственным процессам и приобрели широкое распространение, так как философия TQM отражает новые тенденции в развитии общества.

Отличительной особенностью системы TQM является ее гуманитарная направленность и ориентация на управление человеческими ресурсами.

Основные принципы TQM :

1. ориентация на потребителя,
2. процессный подход для достижения результатов,
3. системный подход к управлению,
4. постоянное улучшение процессов и результатов,
5. вовлечение работников,
6. принятие решений на основе фактов.

Для подтверждения высокого качества разрабатываемого программного обеспечения компании и организации проходят сертификацию по стандарту качества ISO 9000, которая гарантирует, что данная компания или организация выполняет программные продукты в срок и с высоким качеством.

8.4. Модели процесса разработки программного продукта

Процесс разработки программного продукта требует определенной организации. Для этого были разработаны *методы проектирования*. Наиболее распространенными моделями процесса разработки программ являются:

- "водопадная" модель;
- "спиральная" модель.

"Водопадная" модель выдвигает следующие требования к процессу разработки программ:

- Строгое соблюдение последовательности этапов: новый этап не начинается, пока не закончен предыдущий.
- Фиксация в документации и контроль независимыми экспертами результатов каждого этапа.

•Возврат к предыдущим этапам при обнаружении ошибок в строгой обратной последовательности.

Наряду с очевидными достоинствами "водопадная модель" имеет *недостатки*, наиболее ощутимый заключается в невозможности организации доступа к общим данным и распараллеливания процесса разработки программного продукта.

Особенность *"спиральной модели"* заключается в том, что "степень готовности" продукта считается возрастающей величиной в зависимости от времени и этапа разработки. Функция возрастания " степени готовности" изображается в полярной системе координат в виде спирали. "Спиральная" модель по сравнению с "водопадной" моделью имеет *достоинства*:

-предусмотрено быстрое прототипирование будущей программы

-предусмотрена разработка программ с созданием последовательных версий.

На практике удобнее применять *комбинированную модель* процесса разработки, когда за основную модель принимается "спиральная", в которой этапам разработки сопоставляются соответствующие этапы "водопадной " модели.

8.5. Структурное программирование

Для создания "хорошей" программы появляется необходимость придерживаться определенных принципов или определенной дисциплины программирования. Значительный прогресс в области программирования достигается с использованием так называемого *структурного программирования*.

Появление новой технологии, или, как еще говорят, дисциплины программирования, основанной на структурном подходе, связано с именем известного голландского ученого Э.Дейкстры (1965 г.). В своих работах он высказал предположение, что оператор GOTO может быть исключен из языков программирования и что квалификация программиста обратно пропорцио-

нальна числу операторов GOTO в его программах. Такая дисциплина программирования упрощает и структуризирует программу.

Однако представление о структурном программировании, как о программировании без использования оператора GOTO, является ошибочным. Например, Хоор определяет структурное программирование как "систематическое использование абстракции для управления массой деталей и способ документирования, который помогает проектировать программу".

Структурное программирование можно толковать как "проектирование, написание и тестирование программы в соответствии с заранее определенной дисциплиной".

Структурный подход к программированию как раз и имеет целью снижение трудоемкости всего процесса создания программного обеспечения от технического задания на разработку до завершения эксплуатации. Он означает необходимость единой дисциплины на всех стадиях разработки программы. В понятие структурного подхода к программированию обычно включают *нисходящие методы разработки программ (принцип “сверху вниз”)*, собственно структурное программирование и так называемый сквозной структурный контроль.

Основной целью структурного программирования является уменьшение трудностей тестирования и доказательства правильности программы. Это особенно важно при разработке больших программных систем. Опыт применения методов структурного программирования при разработке ряда сложных операционных систем показывает, что правильность логической структуры системы поддается доказательству, а сама программа допускает достаточно полное тестирование. В результате в готовой программе встречаются только тривиальные ошибки кодирования, которые легко исправляются.

Структурное программирование улучшает ясность и читабельность программ. Программы, которые написаны с использованием традиционных методов, особенно те, которые перегружены операторами GOTO, имеют хаотичную структуру. Структурированные программы имеют последовательную

организацию, поэтому возможно читать такую программу сверху донизу без перерыва.

Наконец, структурное программирование призвано улучшить эффективность программ.

К основным *методам структурного программирования* относится, прежде всего, отказ от бессистемного употребления оператора GOTO и преимущественное использование других структурированных операторов, методы нисходящего проектирования разработки программы, идеи пошаговой детализации и некоторые другие соглашения, касающиеся дисциплины программирования.

Всякая программа, в соответствии с структурным подходом к программированию, может быть построена только с использованием трех основных типов блоков.

1. *Функциональный блок*, который на блок-схеме изображается в виде прямоугольников с одним входом и одним выходом:

Функциональному блоку в языках программирования соответствуют операторы ввода и вывода или любой оператор присваивания.

В виде функционального блока может быть изображена любая последовательность операторов, выполняющихся один за другим, имеющая один вход и один выход.

2. *Условная конструкция*. Этот блок включает проверку некоторого логического условия (P), в зависимости от которого выполняется либо один (S_1), либо другой (S_2) операторы:

На языке TP:

if <условие> then <оператор1> else <оператор2>;

3. *Блок обобщенного цикла*. Этот блок обеспечивает многократное повторение выполнения оператора S пока выполнено логическое условие P:

На языке TP циклы реализуются 2 способами. Цикл с параметром:

for <параметр> := <нач.значение> to <кон.значение>;

. . begin

<оператор1>

. . <оператор2>

. . ...

. end;

Цикл с предусловием:

. while <условие>

. .begin

<оператор1>

. . <оператор2>

. . ...

. end;

При конструировании программы с использованием рассмотренных типов блоков эти блоки образуют линейную цепочку так, что выход одного блока подсоединяется ко входу следующего. Таким образом, программа имеет линейную структуру, причем порядок следования блоков соответствует порядку, в котором они выполняются.

Такая структура значительно облегчает чтение и понимание программы, а также упрощает доказательство ее правильности. Так как линейная цепочка блоков может быть сведена к одному блоку, то любая программа может, в конечном итоге, рассматриваться как единый функциональный блок с один входом и одним выходом.

При проектировании и написании программы нужно выполнить обратное преобразование, то есть этот блок разбить на последовательность подблоков, затем каждый подблок разбить на последовательность более мелких блоков до тех пор, пока не будут получены "атомарные" блоки, рассмотренных выше типов. Такой метод конструирования программы принято называть нисходящим ("сверху вниз").

При нисходящем методе конструирования алгоритма и программы первоначально рассматривается вся задача в целом. На каждом последующем этапе задача разбивается на более мелкие подзадачи, каждая подзадача, в ко-

нечном итоге на еще более мелкие подзадачи и так до тех пор, пока не будут получены такие подзадачи, которые легко кодируются на выбранном языке программирования. При этом на каждом шаге уточняются все новые и новые детали ("пошаговая детализация").

В процессе нисходящего проектирования сохраняется строгая дисциплина программирования, то есть разбиение на подзадачи осуществляется путем применения только рассмотренных типов конструкций (функциональный блок, условная конструкция, обобщенный цикл), поэтому, в конечном итоге, получается хорошо структурированная программа.

В настоящее время в помощь структурному подходу к программированию появилось новое направление - *объектное программирование* , не отменяющее, а дополняющее принципы структурного подхода, позволяющее разрабатывать и модернизировать программный код с меньшими затратами времени, распределять задачу программирования между большим количеством программистов.

8.6. Основные принципы структурной методологии

1. *Принцип абстракции***.** Этот принцип предполагает рассмотрение всей программной систем, как многоуровневые системы. Каждый уровень является детализацией предыдущих.

2. *Принцип формальностей***.** Каждая программа должна реализовывать некоторый алгоритм, который построен на определенной математической модели решения задач. Математическая модель содержит математические зависимости и формы, а также ограничения , накладываемые на исходные данные. При этом формулы предназначаются для преобразования исходных данных в результат.

3. *Принцип "разделяй и властвуй".* Определяет способ решения трудных задач путем разделения этой задачи на множество мелких, легко решаемых. Этот принцип реализуется путем создания подпрограмм.

4. *Принцип модульности*. Определяет способ создания больших программ, у которых возникают проблемы с оперативной памятью, в следствии их большого объема или большого объема обрабатываемых ими данных. Принцип модульности ускоряет создание больших программ за счет использования ранее созданных описаний.

5. *Принцип открытости*. Программы должны быть открытыми для быстрых модификаций, поэтому они должны быть понятны и хорошо откомментированы. Использование структурной методологии позволило повысить надежность и эффективность программ, а также сократить сроки создания и сопровождения программ.

Широко используемым примером структурной технологии является технология проектирования "сверху вниз". Ее суть заключается в постепенной, пошаговой детализации , выполняемой программы, при этом на каждом этапе детализации должны быть рассмотрены альтернативные варианты решения и выбранные наилучшие.

8.7. Модульное программирование.

В качестве описаний могут выступать описания переменных, констант, процедур и функций.

Использование модулей позволяет сократить объём программы, разделяя её на части, а именно: одна – главная программа, а остальная – модули.

Модули могут отдельно компилировать, но не могут выполнять.

В Turbo Pascal можно использовать стандартные модули или разрабатывать пользовательские модули.

Модульное программирование, т.е. разработка программ, которые представляют из себя главную программу и набор модулей, является наиболее оптимальным процессом для сложных программных систем.

8.8. Объектно-ориентированное программирование (ООП)

По мере прогресса в области программирования и информационных технологий стал развиваться новый подход, согласно которому программа должна быть моделью предметной области. Само программирование должно быть моделированием процессов, происходящих в этой предметной области. Этот подход называется *объектно-ориентированное программирование* (ООП). Он был разработан с целью улучшения процесса создания больших программных комплексов за счет снижения их сложности и стоимости. В процессе ООП программист руководствуется привычными понятиями той предметной области , для которой создаются программы. Эти понятия в программе отображаются в программе как объекты.

Объект- это совокупность данных и подпрограмм, предназначенных для работы с этими данными.

Подпрограммы, определяемые для конкретного объекта, называются *методами*. Таким образом, характерной чертой объектов является *инкапсуляция*, то есть, объединение данных и методов их обработки.

Любая ООП состоит из двух частей: описание объектов и последовательности действий, связанных с передачей сообщений между этими объектами. Рассмотрим объект точка на экране.

Программа написанная на языке Turbo Pascal.

```
type
point =object;
x,y : integer;
invisbe : boolean;
procrdure create ( a,b :integer);
procedure switchon ;
procedure switchoff;
procedure mov (dx,dy :integer);
function getxx :integer;
function getyy :integer;
```

```
end;
procedure point.create (a,b :integer);
begin x:=a;
y:=b:
visible:=false;
end;
procrdure point.switchon;
begin
visible:=true;
putpixel (x,y,getcolor);
end;
.....
function point.getxx :integer;
begin
getxx:=x;
end;
.....
{Конец описания объекта. Дальше начинается текст программы.}
var
onepoint.create :point; {экземпляр объекта}
.....
begin
onepointcreate (100,200);
onepoint.switchon;
.....
end.
```

Методы предназначены для обработки только тех данных, которые инкапсулированы с этими данными в объект.

Для работы с объектами необходимо указать в программе экземпляры объектов в разделе описания переменных. Для обращения к методам соответ-

ствующего объекта требуется использовать составное имя: имя экземпляра объекта. имя метода.

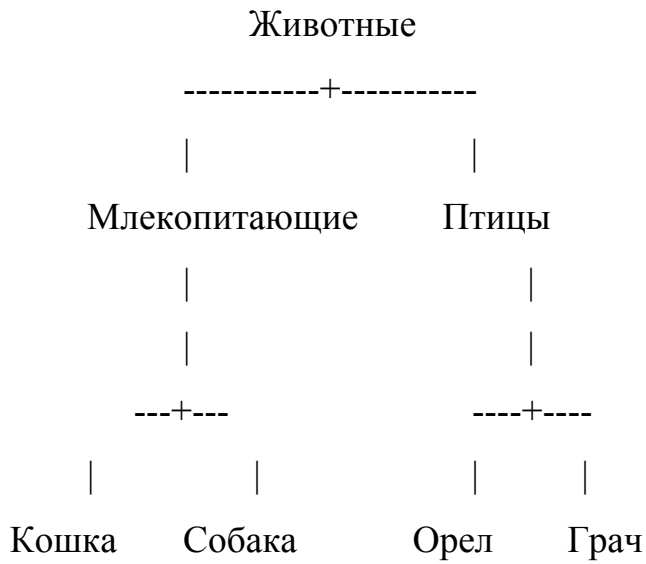
При описании подпрограмм, реализующих методы объекта, обращение к полям объекта записывается без составного имени.

Например, поле x:=a и y:= b в процедуре point . create.

С помощью сообщений объекты активно воздействуют друг на друга, как бы моделируя действия людей, обменивающихся информацией. Создавая программы, использующие методологии ООП, программист моделирует не в виде бессистемной совокупности, а как набор независимых объектов с полностью определенным поведением, задаваемым методами.

8.8.1. Наследование и полиморфизм

Потенциал ООП проявляется только тогда, когда все объекты, описываемые в программе, расположить в виде иерархической структуры, в которой отражены наследования свойств родительских, т.е. выше расположенных объектов, дочерними.



Наследование – это такое решение между объектами, когда один дочерний объект повторяет структуру и поведение другого объекта- родителя.

Для описания объектов- потомков используется также структура, которая описывает объект, в котором после объекта идет имя родителя.

Пример: объект-точка на экране.

type

```
point1=object(point);
procedure create;
end;
procedure point1.create;
begin
x:=10;
y:=10;
end;
```

Объект point1 наследует все методы, т.е. подпрограммы, описанные для объекта point.

Полиморфизм – это свойство различных объектов, выполнять одно и то же действие по своему.

В объекте point1, также как и в объекте point описана процедура создания точки create , но она реализована по другому, в этом и заключается полиморфизм. Т.о. при ООП необходимо как можно более полно описать структуру и поведение объектов. Любой объект может быть объявлен потомком ранее описанного объекта. в этом случае он наследует все данные и методы объекта- родителя, но может дополнять их своими данными и методами или переопределять методы объекта- родителя.

8.8.2. Общие принципы ООП

1. Все является объектом. Объект характеризуется своими атрибутами, которые создаются переменными и несут информацию об этом объекте. В объекте также описываются модели его поведения или методы его обработки информации.

2. Вычисление осуществляется путем взаимодействия или обмена данными между объектами, при котором один объект идет на выполнение некоторого действия.

3. Каждый объект является представлением некоторого класса объектов, который выражает общие свойства этих объектов, включенных в класс.

4. Каждый объект имеет независимую память , которая состоит из свойств его объектов- родителей.

5. Каждый объект- потомок может переопределить проявление своего поведения.

6. Объекты, определенные в иерархическую структуру с общим корнем, описывают предметную область и называются схемой иерархии наследования.

Тема 9. Базы данных –2 часа

9.1. Концепция баз данных

Традиционно фиксация данных осуществляется с помощью конкретного средства общения (например, с помощью естественного языка или изображений) на конкретном носителе (например, камне или бумаге). Обычно *данные* (факты, явления, события, идеи или предметы) и их *интерпретация* (*семантика*) фиксируются совместно, так как естественный язык достаточно гибок для представления того и другого.

Примером может служить утверждение "Стоимость авиабилета 128". Здесь "128" - данное, а "стоимость авиабилета" - его семантика.

Активная деятельность по отысканию приемлемых способов обобществления непрерывно растущего объема информации привела к созданию в начале 60-х годов специальных программных комплексов, называемых *системами управления базами данных (СУБД)*.

Основная особенность СУБД - это наличие процедур для ввода и хранения не только самих данных, но и для описания их структуры. Файлы, снабженные описанием хранимых в них данных и находящиеся под управлением СУБД, стали называть банками данных, а затем базами данных (БД).

База данных – эта поименованная хранимая в виде записей в долговременной памяти совокупность структурированных данных о предметной области.

9.2. Технология баз данных

Последние три десятилетия в области разработки информационных систем сформировалась новая информационная технология – *технология БД*. Она включает в себя методы, средства, способы и практические рекомендации применения БД в качестве основного компонента в различных информационных системах (ИС). Существуют *разновидности информационных систем* , использующие технологию БД:

1.документальные (ДС)

2.фактографические (ФС).

Документальные системы предназначены для работы с документами на естественном языке: книги, тезисы, статьи... Наиболее распространенным видом ДС являются *информационно-поисковые системы* (*ИПС*), которые предназначены для накопления и поиска по различным критериям документов. В состав ИПС входят: программные средства, поисковый массив документов и средства поддержки информационного языка этой системы.

Программные средства ИПС служат для организации ввода и хранения информации, а также обработки пользовательских запросов на поиск документов. Поисковый массив документов обычно называют БД, и он не содержит текстов, а включает только библиографические сведения: рефераты и аннотации. При выполнении поиска документа используют поисковые образы, отражающие содержание документа. Эти образы строятся путем индексирования документов и являются ключевыми словами поискового документа.

Информационные языки (ИЯ) в ИПС строятся на базе ограниченных естественных языков. Это приводит к неоднозначной интерпретации смысла запросов. Для решения этой проблемы вводятся словари - тезаурусы. Они описывают лексические единицы языка и смысловые связи между ними. Наибольшим распространением для реализации таких ИЯ получили языки дескрипторного типа, которые в качестве лексических единиц или дескрип-

торов используют слова или словосочетания, соответствующие основным понятиям предметной области. Дескрипторы заносятся в тезаурус ИПС.

Информационный запрос формируется на таком ИЯ, затем преобразуется в поисковый образ запроса (ПОЗ), ПОЗ сопоставляется с поисковыми образами документов по критерию смыслового соответствия, и в результате выдаются все документы, которые оказались релевантными этому запросу.

Характерной особенностью *фактографических систем* является то, что они работают не с текстом, а с фактическими сведениями, которые представлены в виде записей. Основные компоненты ФС – это сами БД и системы управления БД (СУБД). На базе ФС создаются справочники, системы анализа и управления предприятиями, бухгалтерские системы.

В дальнейшем будем рассматривать фактографические системы.

СУБД должна предоставлять доступ к данным любым категориям пользователей, включая и тех, которые практически не имеют или не хотят иметь представления о:

-физическом размещении в памяти данных и их описаний;

-механизмах поиска запрашиваемых данных;

-проблемах, возникающих при одновременном запросе одних и тех же данных многими пользователями (прикладными программами);

-способах обеспечения защиты данных от некорректных обновлений и (или) несанкционированного доступа;

-поддержании баз данных в актуальном состоянии и множестве других функций СУБД.

9.3. Проектирование баз данных

Проект базы данных надо начинать с анализа информационных потоков и документооборота, определения информационных объектов предметной области, функций по их обработке, связей между ними и выявления требований к ней отдельных пользователей (сотрудников организации, для которых создается база данных).

Основными этапами проектирования БД являются :

1.Инфологическое проектирование.

2.Логическое проектирование и выбор инструментальных средств СУБД

9.3.1. Инфологическое проектирование

В результате *инфологического проектирования* БД должна быть создана инфологическая модель. Эта модель строится путём анализа и определения объектов и связей между ними в предметной области.

Существует *функциональный* и *объектный подход* в инфологическом проектировании.

Функциональный подход реализует принцип “от задач”. *Объектный подход* не фиксирует количество решаемых задач, а включает в инфологическую модель только объекты и связи между ними. Смешанный подход объединяет предметный и функциональный.

Рассмотрим *инфологическое проектирование для объектного подхода*. Для описания инфологической модели здесь используются диаграммы “объекты - связи” или, по-другому, ER - диаграммы.

Описание предметной области предполагает рассмотрение ее в качестве совокупности связанных реальных сущностей, которые имеют набор свойств или атрибутов, характеризующих только эту сущность. Группы всех подобных сущностей образуют объект, в котором фиксируются только значимые для данной ИС свойства. Связь представляет собой отношение между двумя объектами.

Объект должен иметь один или несколько ключевых атрибутов, т.е. таких атрибутов, которые однозначно или уникально определяют экземпляр объекта среди всех других экземпляров объектов указанного типа.

Совокупность объектов и связей между ними определяет структуру предметной области, на основе которой строится инфологическая модель.

Каждому объекту можно сопоставить экземпляр объекта, а тот в свою очередь образуется совокупностью конкретных значений атрибутов, и должны однозначно определяться с помощью ключа.

Ключ – один или несколько атрибутов, которые однозначно определяют экземпляр объекта.

Совокупность атрибутов объекта должна отвечать определенным требованиям (*требованиям нормализации*). Эти требования позволяют на этапе инфологического проектирования строить базы данных, обеспечивающие целостность данных и устраняющие дублирование:

1. Все атрибуты объекта кроме ключевых, называют описательными, и они должны функционально зависеть от ключа.
2. Между описательными атрибутами не должно быть функциональной связи.
3. Между ключевыми атрибутами не должно быть функциональной связи.
4. Каждый описательный атрибут не должен зависеть от ключа транзитивно, т.е. через другой промежуточный атрибут.
5. В случае транзитивной зависимости между атрибутами нужно выполнить расщепление совокупности атрибутов с целью образования двух информационных объекта вместо одного.

9.3.2. Логическое проектирование

Каждая система СУБД поддерживает ту или иную модель данных. *Логическая модель данных* определяет правила порождения допустимых видов структур данных и возможные операции над ними. Основной целью проектирования БД является решение проблемы выбора оптимальной логической структуры для заданного набора данных.

Различают:

иерархические модели данных,

сетевые модели данных,

реляционные модели данных.

Иерархические и сетевые модели – это разновидность графовых моделей. В иерархических моделях связи между данными определяются в виде дерева графа, при этом связи односторонние: от предка к потомку. Основными операциями в иерархических моделях являются поиск записи с предварительным перемещением по дереву и перемещение сверху вниз.

Сетевые модели, в отличие от иерархических, реализуют связи всех со всеми. Типичные операции: поиск, замена значений, запись. Сетевые модели являются улучшенной иерархической моделью.

9.4. Проектирование реляционной базы данных

Впервые *реляционную модель данных* (РМД) предложил ученый Кодд, который упростил структуру отображаемой БД, исключив непосредственные указатели на предков и потомков.

Для отображения информационных объектов и связей Кодд предложил использовать отношения, представляемые на уровне пользователя в виде таблиц, и математические операции над ними. РМД наиболее проста и имеет в основе развитый математический аппарат, поэтому она фактически стала стандартной моделью представления данных в СУБД .

9.4.1. Основные понятия реляционной алгебры.

Операндами реляционной алгебры являются постоянные и переменные отношения, которые представляются как подмножество декартового произведения списка доменов.

Под *доменом* будем понимать некоторое множество значений.

Декартовым произведением доменов D1´D2 ´D3. . . .Dk является множество всех кортежей (V1,V2,V3, . . . ,Vk) длины k таких, что :

V1ÎD1, V2 ÎD2, V3 ÎD3, . . ., Vk ÎDk.

Отношением называется некоторое подмножество декартового произведения одного или нескольких доменов.

Будем считать, что это подмножество конечно. Элементы отношения называются кортежами. Отношение характеризуется арностью, т.е. количеством компонент. Любое отношение легко представить в виде таблицы, где каждая строка - кортеж, а столбец соответствует атрибуту (домену) этого отношения.

С помощью отношений удобно описывать как объекты, так и связи инфологической модели.

Пример:

отношение ГОРОД (название, район, население).

Название	Район	Население
Москва	Центральный	7 000 000
Ульяновск	Поволжье	800 000
Димитровград	Поволжье	180 000

Подмножество атрибутов данного отношения, обладающее свойством уникальности и неизбыточности, называют *первичным ключом* этого отношения, который обеспечивает основной механизм адресации на уровне кортежей.

Список имен атрибутов отношения называют *схемой отношения*. Совокупность схем отношений, используемых для представления информации о предметной области, называется реляционной схемой БД, а текущее значение соответствующих отношений называется реляционной БД .

Связи между отношениями реализуются путем указания одинаковых атрибутов - ключей связи.

Ключом связи является первичный ключ главного отношения.

Ключом связи в подчиненном отношении является либо часть первичного ключа этого отношения, либо атрибут, не входящий в состав первичного ключа этого отношения. Ключ связи в подчиненном отношении называется внешним ключом.

Операции в реляционной модели предназначены для организации запросов к БД в терминах отношений. Эти запросы относятся к включению, соединению, выборке кортежей соответствующих отношений.

Рассмотрим *основные операции над отношениями*.

Пусть R и S - отношения, они интерпретируются таблицами. Арность этих отношений одинакова.

Объединение отношений: *R union S* .

Результатом этого объединения будет отношение, содержащее кортежи, принадлежащие и R и S.

Разность отношений: *R difference S*.

Результатом этой операции будет отношение, содержащее кортежи принадлежащие R, но не S.

Декартовое произведение отношений: *X cross У*.

Пусть X, У - отношения арности k и t соответственно. Тогда X cross У есть отношение, содержащее множество кортежей длины k+t, первые k компонентов которых образуют кортежи отношения X, а последние t компонент - кортежи отношения У.

Пример:

Даны 2 отношения X и У.

Отношение X (Адрес, Улица, Дом) - таблица из 3 столбцов и 3 строк:

Адрес	Улица	Дом
A1	B1	C1
A2	B2	C2
A3	B3	C3

Отношение У (Квартира, Подъезд, ФИО):

Квартира	Подъезд	ФИО
D1	G1	E1
D2	G2	E2
D3	G3	E3

X cross У - таблица из 6 столбцов и 6 строк:

Адрес	Улица	Дом	Квартира	Подъезд	ФИО
A1	B1	C1	D1	G1	E1
A1	B1	C1	D2	G2	E2
A2	B2	C2	D1	G1	E1
A2	B2	C2	D2	G2	E2
A3	B3	C3	D1	G1	E1
A3	B3	C3	D2	G2	E2

Проекция Proj R(d1,d2 . . . dn).

В отношении R арности k, (k>n) проекция получается путем выделения проецируемых доменов d1, d2 . . . dn в отдельное отношение и удаления повторяющихся кортежей.

Пример:

PROJ X cross У (Адрес, Дом)

Адрес	Дом
A1	C1
A2	C2
A3	C3

Селекция SEL (F) R. Пусть F – формула, образованная операндами-доменами, арифметическими операциями сравнения и логическими операциями AND, OR, NOT. В этом случае селекция SEL (F) R есть множество кортежей t, принадлежащих , таких R, для которых формула F является истинной.

Соединение R join S.

Результатом будет отношение, содержащее конкатенацию (т.е.со-единение) каждого кортежа отношения R с кортежем отношения S, у ко-торых совпадают значения в общем домене.

9.4.2. Правила Кодда, которым должна соответствовать РБД:

1. *Правило информации*. Вся информация, хранящаяся в БД находится в таблицах.

2. *Правило гарантированного доступа*. Доступ к информации и значениям в таблице гарантированно осуществляется, если указать имя таблицы, столбца в таблице и значение первичного ключа в таблице.

3. *Правило поддержки недействительных значений*. В качеств отсутствующих значений БД может использовать NULL - значения. Значения ключевых атрибутов не могут быть NULL - значения.

4. *Правило обновления, добавления, удаления*. Эти правила реализуются на уровне записей таблицы.

5. *Правило независимости физических данных*. При изменении способов хранения данных и доступа к ним на уровне физической организации логическая структура РБД не меняется.

6. *Правило использования условия целостности данных*. РБД должна поддерживать возможность определения условия целостности данных.

9.5 Язык SQL

SQL (Structured Query Language) был разработан с целью создания унифицированного средства извлечения информации из БД и организации связи между различными СУБД

Основные возможности современного SQL:

1) Организация данных (определение структуры и связей).

2) Чтение или извлечение данных.

3) Модификация данных.

4) Управление доступом данных.

5) Совместное использование данных.

6) Обеспечение целостности данных.

7) Может использоваться как средство разработки приложений.

8) SQL может быть встроен в систему программирования и СУБД.

Имеется *четыре группы операторов SQL*:

•*Операторы определения данных* – позволяют создавать, удалять таблицы и изменять их определение. Операторы манипуляции данными - удаления, добавления и нахождения записей.

•*Операторы контроля над данными* предоставляют средства для обеспечения безопасности.

•*Операторы завершения транзакций* предоставляют возможность принятия или отмены изменений в случае возникновения ошибки.

9.5.1. Операторы описания данных.

CREATE SCHEMA - создание схемы

CREATE TABLE - создание таблицы

CREATE VIEW - создание формы

CREATE DOMAIN - создание столбца

9.5.2. Операторы манипулирование данными.

SELECT - выбор данных

INSERT - вставка

UPDATE - обновление

DELETE - удаление

Наиболее часто используемым в языке SQL является оператор SELECT. Этот оператор предназначен для чтения данных из БД и формирования в качестве результата таблицы, содержащей выбранные данные.

Оператор SELECT может содержать 6 основных подразделов, различающихся ключевыми словами

SELECT (выбрать)

ALL (все столбцы) или

* (все столбцы) или

< имя одного или нескольких столбцов >

DISTINCT (обозначает удаление повторяющихся строк)

FROM < из каких таблиц >

WHERE <условие поиска>

GROUP BY < имя одного или нескольких столбцов > (Используется для вывода данных, сгруппированных по определенному признаку)

HAVING < условие> (Условие позволяет отбирать группы)

ORDER BY < столбец> (Отсортировать данные по указанному столбцу)

Тема 10. Телекоммуникации – 2 часа

10.1. Локальные сети

Локальные сети представляют основу для создания принципиально новой индустрии обработки данных.

Пользователи компьютеров, объединенных в локальную сеть могут передавать друг другу сообщения, совместно использовать базы данных или устройства (например, принтеры), что значительно повышает удобство и эффективность коллективного труда.

Локальные сети представляют собой систему распределенной обработки данных, охватывающих небольшие территории внутри отдельных предприятий, учреждений, контор и имеет общий канал, через который производится передача сообщений. Для объединения компьютеров в локальную сеть, компьютеры необходимо соединить специальными проводами (кабелями) и вставить в них платы сетевых адаптеров, позволяющих предавать информацию по кабелям сети (имеются, впрочем, сети, в которых обмен информации осуществляется без специальных проводов - с помощью радиоволн или через осветительную сеть). Однако одних проводов и сетевых адаптеров недостаточно, необходимы программы, обеспечивающие обмен информацией по локальной сети.

При небольшом количестве компьютеров, объединенных в локальную сеть, может оказаться целесообразным использование одноуровневой сети, в ней все компьютеры сети равноправны, все применяются для работы пользо-

вателей , и каждый пользователь сам определяет, какие ресурсы на его компьютере доступны другим пользователям через локальную сеть .

Обычно компьютеры при этом работают под управлением операционной системы Windows NT , но на каждом компьютере запускается специальный драйвер , обеспечивающий доступ к локальной сети .Примерами таких сетевых программ Lantastic, Personal Netware и др.

Однако при большом количестве компьютеров в локальной сети или при особых требованиях к доступности и сохранности, совместно используемых данных требуются более мощные сети. Обычно в них один или несколько компьютеров выделяются исключительно для обслуживания локальной сети и совместно используемых ресурсов (данных и устройств). Такие компьютеры называются серверами или файл-серверами, как правило, это достаточно мощные компьютеры на основе микропроцессоров Pentium II или III с большим количеством оперативной и дисковой памяти. На дисках серверов располагаются совместно используемые программы, базы данных и т.д. Остальные компьютеры локальной сети часто называются рабочими станциями. На тех рабочих станциях, где требуется обрабатывать только данные на сервере, часто для экономии даже не устанавливают жестких дисков.

На рабочих станциях такой локальной сети может использоваться любая операционная система , например Windows NT , и должен быть запущен драйвер, обеспечивающий доступ к локальной сети . Но на серверах локальной сети обычно используются другие операционные системы : там требуется обеспечивать одновременную обработку многих запросов от рабочих станций , к чему Windows NT как однопользовательская и однозначная система не приспособлена .

На компьютерах типа IBM PC была популярна сетевая операционная система фирмы Novell , но используются в настоящее время различные варианты UNIX , OS/2 , Banyan Vines , Windows и т.д. Эти операционные системы предоставляют мощные средства разграничения доступа к данным, обеспече-

ние целостности и сохранности совместно используемых баз данных и множество удобных сервисных возможностей.

Локальные сети *классифицируются по двум основным сферам использования* :

1). Автоматизация информационно-учрежденческого дела ,

2). Управление промышленными процессами .

К *первому типу* относятся также сети для школ , больниц и т.д. , а *ко второму* – локальные сети сложных транспортных средств , автоматизации научных экспериментов , тренажеров и т.д.

Основное отличие этих двух типов сетей заключается в том , что для локальных сетей второго типа , как правило , требуется обеспечить гарантированное время доставки сообщений , а для локальных сетей первого типа таких строгих требований нет . В основу реализации сетей первого типа положено использование физической среды в режиме случайного разделения времени .

Однако локальные сети не могут полностью удовлетворить все нужды в обмене информацией между компьютерами . Они потому и называются локальными (от латинского locus - место) , что связывают компьютеры , находящиеся близко друг от друга . Однако компьютеры и локальные сети можно связывать между собой с помощью каналов связи (телефонной , радиорелейной , спутниковой и т.д.) ; образуя распределенные вычислительные системы и сети различного назначения .

Существуют различные способы соединения компьютеров в сеть . Существуют *3 основных вида* : " звезда " , " кольцо " и " магистраль " .

10.2. Глобальная сеть – Интернет

InterNET – это общемировая совокупность компьютерных сетей, связывающая между собой миллионы компьютеров. Зародышем её была распределенная сеть ARPAnet, которая была создана в конце 60-х годов по заказу Министерства Обороны США для связи между собой компьютеров этого мини-

стерства. Разработанные принципы организации этой сети оказались настолько удачными, что многие другие организации (особенно университеты и правительственные учреждения) стали создавать собственные сети на тех же принципах. Эти сети стали объединяться между собой, образуя единую сеть с общим адресным пространством (подобно тому, как все телефонные станции одного города поддерживают единую систему телефонных номеров). Эта единая сеть (или сеть сетей, совокупность сетей) и стала называться InterNET.

Более 90% информации Интернет в настоящее время представлено на английском языке. Причина - более 60% сетевых ЭВМ, установленных в Интернет, находится в США. Там же находятся основные производители компьютеров, разработчики программного обеспечения и телекоммуникационного оборудования.

Информация на русском языке в Интернет составляет сотни миллионов страниц, размещенных на более 200 тысячах сетевых ЭВМ. Но пока это всего около 1% от ресурсов Интернет в развитии бизнеса, науки, экономики и образования - основных двигателей прогресса.

Сегодня 1,5 миллиона (1%) россиян являются активными пользователями Интернет. Согласно новой федеральной программе информатизации к 2005 г. более 10 миллионов россиян в системе образования Российской Федерации приобретут навыки работы в Интернет.

Работа в Интернет требует, прежде всего, умения работать на персональных компьютерах. Наиболее распространенной моделью персональных компьютеров являются вычислительные машины IBM PC, созданные фирмой IBM и выпускаемые в различных странах мира.

Для усвоения навыков читать, писать и искать информацию на компьютерах IBM PC необходимо освоить возможности операционной системы Windows, редактора текстов Word, а также сетевого браузера Internet Explorer.

Для получения информации в сети Интернет необходимо освоить правила навигации в Интернет, работы с информационными сайтами, электронной почтой и поисковыми системами.

Для эффективного поиска информации в Интернет необходимо освоить языки запросов к поисковым системам и базам данных на ЭВМ, а также уметь читать и интерпретировать результаты работы ЭВМ и поиска информации на русском и английском языках.

С помощью Интернет уже в ближайшее время станет широко доступна электронная коммерция - возможность заказа самых различных товаров - книг, билетов, подарков, компакт-дисков и т.п. - с помощью сети электронных магазинов, доступных в сети уже сейчас.

10.3. Сетевые протоколы

Для взаимодействия между собой программы в Интернете используют *протоколы*. *Протокол* – это набор правил и соглашений, используемых при передаче данных или коммуникациях. Возможно, вам знаком термин "протокол" в контексте международных дипломатических отношений. Рукопожатие - одна из форм приветствия - это протокол, принятый во многих культурах. Это протокол, которым пользуются, чтобы поздороваться. Компьютеры в сети тоже общаются друг с другом, используя общий язык - протоколы связи, без которых было бы немыслимо управлять сетью и передавать данные. Другими словами каждая программа, претендующая на работу в сети, должна следовать определенным правилам для приема и передачи данных.

HTTP – Hyper Text Transfer Protocol – это протокол, предназначенный для передачи гипертекста . *Гипертекст* - это такой текст , в котором содержатся ссылки на другой текст . Все справочные системы по ТР основана на гипертексте . Гипертекстовые файлы называют HTML - файлы .

FTP – протокол для передачи файлов (программных) File Transfer Protocol.

Клиенты и серверы

Модель *клиент - сервер* является одной из сетевых технологий. *Клиент* запрашивает услугу или информацию, а *сервер* отвечает на запрос клиента и передает информацию. Таким образом, приложение - клиент выдает запросы, а приложение сервер на них отвечает. Примером сервера может служить система WAIS - обеспечивающая доступ к компьютерным базам данных по всему земному шару. Эта система предлагает список баз данных , позволяет выбрать среди них нужную и осуществляет в ней поиск . Поиск осуществляется по нужной для вас теме.

Для каждого приложения, будь то *WWW* (*WWW - World Wide Web*, Всемирная Паутина, – часть Интернет, основанная на протоколе HTTP), обмен новостями или просмотр видеофильма, существует определенный стандартный протокол. Следовательно, пользователи Сети могут быть уверены в том, что их сообщения правильно поймет и обработает практически любой другой компьютер в Интернете.

10.4. Службы Интернет

Службы Интернет – это подразделения и программное обеспечение, созданные для того, чтобы облегчить работу в Интернет: сделать её более простой и удобной.

К *программному обеспечению* можно отнести те программы, которые мы непосредственно используем при работе в сети. Такие как Internet Explorer, Natscape Navigator, а также множество других вспомогательных программ (arhivrar, AVP, DJVU и т.д.).

К подразделениям относятся ресурсы Интернет, такие как WWW - ресурсы, FTP - ресурсы и т.д.

Службы Интернет:

- браузеры;
- поиск информации в Интернет;
- электронная почта;

- телеконференции;
- получение файлов по FTP.

10.4.1. Браузеры

Основными инструментами для работы в Интернет являются *браузеры.* Наиболее популярны браузер Netscape Navigator фирмы Netscape и браузер Internet Explorer фирмы Microsoft для операционной системы Windows.

Для входа в Интернет необходимо запустить браузер, имеющийся на компьютере, указав в окне имя сайта. В результате запуска на экран выводится картинка (Web - страница), похожая на экран встроенного справочника Windows - программы. На Web - странице могут находиться надписи, тексты, рисунки и другие объекты. Как и в справочнике Windows - программы, щулкнув мышью любое выделеное (обычно подчеркнутое) слово (ссылку), Вы перейдете к Web - странице, соответствующей этому слову (ссылке). Однако, в отличие от встроенных справочников программ, выделенная по ссылке Web - страница может находиться на любом другом Web - сервере (то есть может быть передана Вам уже другим Web - сервером). Щелкая по разным ссылкам, Вы можете в поисках информации за несколько минут проскакать по Web - серверам в десятке разных стран.

Ссылки могут быть на гипертексты, как хранящиеся на том же компьютере, так и размещенные на других ЭВМ. Гипертексты внутри сайтов имеют внутреннюю адресацию, похожую на адресацию файлов в операционных системах. Эти адреса имеют доменную структуру, указываются в окне браузера "Адрес" и называются указателями гипертекстовых ресурсов (URL).

Пример доменного имени указателя ресурсов URL:

http: //pmi.ulstu.ru , где HTTP - это протокол, предназначенный для передачи гипертекста , pmi - название подразделения в организации ulstu , ulstu - организация , ru - код страны.

www.ya.ru , где www - ресурс глобальной сети , ya - в данном случае поисковая система , ru - код страны .

Для поиска серверов и сайтов в Интернет используются электронные каталоги, создаваемые и подерживаемые крупнейшими порталами. В русском секторе Интернет крупнейшими порталами являются:

www.List.ru- электронный каталог сайтов в Интернет;

www.Port.ru- крупнейший телекомуникационный портал;

www.infoart.ru- крупнейший информационный портал.

Обратившись к этим каталогам, можно найти сайты по многим отраслям – экономика, образование, банки, Интернет, наука, культура и т.д. Кроме того, в этих электронных каталогах можно найти рейтинги самых посещаемых сайтов и серверов в Российском секторе Интернет.

10.4.2. Поисковые системы

Для предметного поиска в Интернет сайтов и гипертекстов по заданной теме используются *поисковые системы*. Одной из наиболее удобных среди них является система

www.rambler.ru – система поиска информации в Интернет.

Для поиска информации в окне этих систем указываются ключевые слова. Пример запросов с ключевыми словами:

"электронный учебник информатики";

"музыка mp3".

Результатом поиска будет перечень ссылок на гипертексты с краткими аннотациями, в которых находятся сведения:

а) об электронных учебниках по информатике,

б) о музыкальных сайтах.

В частности, информация об электронной версии учебника информатики в сети Интернет и обо всех электронных учебниках, имеющихся в российском секторе Интернет.

Так же, среди отечественных систем так же очень популярны Апорт и Яндекс:

www.aport.ru;

www.yandex.ru.

Особенность отечественых поисковых систем - они ищут информацию с русскоязычными текстами по запросам на русском языке. В зарубежных поисковых системах информация и запросы, как правило, вырожаются на английском языке.

Среди международных информационно-поисковых систем наиболее известны поисковые системы AltaVista, Infoseek, Yahoo. Эти системы позволяют выражать запросы и искать информацию не только на английском, но и на русском, а также испанском, французком, немецком и других языках мира:

www.altavista.com - поисковая система AltaVista,

www.yahoo.com - поисковая система Yahoo.com,

www.infoseek.com - поисковая система Infoseek.com

Поиск информации в сети Интернет всеми этими системами производится по запросам. Однако некоторые из поисковых систем дополнительно позволяют искать информацию по тематическим каталогам, как это делается в обычных библиотеках.

Запросы на поиск информации в Интернет могут состоять из одного или нескольких слов на русском или английском языке.

Примеры запросов:

запрос: Интернет;

запрос: учебник информатики;

запрос: computer science.

Ответы на запросы – ссылки на сайты и гипертексты, в которых имеются указанные ключевые слова, а также аннотации, которыесинтезируют (попросту - "выдумывают") поисковые системы. Эти ответы могут содержать от двух - трех до нескольких тысяч ссылок. Каждая из таких ссылок указывает на гипертекст, в котором имеются заданные ключевые слова.

Система Апорт и Яндекс вначале указывают гипертексты, в которых ключевые слова найдены в названии, затем в списке ключевых слов самого

гипертекста, а затем в самом гипертексте - вначале внутри абзацев, а уже затем во все тексте в целом.

Некоторые слова и словосочетания могут вовсе отсутсвовать в гипертекстах, хранящихся в Интернет. Например, фраза "глокая куздра" состоит из слов, похожих на слова русского языка, но они не имеют реального смысла и не используются в русском языке.

10.4.3. Электронная почта

Электронная почта – это передача писем в Интернет с помощью персональных компьютеров. Если обычная почта - это пересылка бумажных писем и документов с помощью государственных почтовых служб, то электронная почта - это передача электронных писем с использованием международной сети Интернет.

Для ведения электронной почты используются специальные почтовые программы, которые можно найти в Интернет либо в современных операционных системах. В частности, в операционную систему Windows входит почтовая программа Outlook Express.

Для отправки электронных писем их текст может быть подготовлен с помощью почтовой программы и сразу же отправлен по Интернет. В считанные секунды или минуты она достигнет адресата - электронное письмо будет записано в почтовый ящик по адресу, который указывается при отправке письма:

Пример адресов электронной почты в сети Интернет:

master@mail.ru

olga@rambler.ru

< имя >@hotmail.com

Первое имя в любом адресе электронной почты – *идентификатор владельца* почтового ящика. *Второе имя – адрес почтового сервера*, где хранятся полученные электронные письма.

В частности, *mail.ru* и *rambler.ru* – адреса серверов российских электронных почтовых служб Mail.Ru и Rambler.Ru соответственно, а hotmail.com – адрес сервера международной электронной почтовой службы HotMail.

В этих почтовых службах можно бесплатно открыть электронный почтовый ящик определенного объема и хранить там свои письма и вести переписку по Интернет. Для открытия почтового ящика в адресной строке браузера указывается один из этих почтовых сайтов и производится регистрация нового почтового адреса.

При регистрации нужно указать свою фамилию, имя, отчество и другие данные (но не обязательно эти данные должны быть достоверны, и как показывает практика: большинство ящиков заведено под вымышленными именами), а также указать идентификатор создаваемого почтового ящика и личный пароль (а вот пароль нужно не только ввести, но и запомнить, иначе Ваш ящик будет лишь занимать место на почтовом сервере и его вскоре удалят), по которому почтовая система будет давать доступ к Вашему почтовому ящику.

Личныи идентификатор должен быть уникальным и легко запоминаемым. Уникальность означает, что он должен отличаться от всех остальных имен, зарегестрированых на почтовом сервере. Для имени почтового ящика обычно используются инициалы и фамилия владельца либо название службы или подразделения организации.

Для ведения переписки после регистрации в браузере указывается адрес почтового сервера, а затем указываются личный идентификатор и пароль. Если они указаны правильно, то Вы получаете доступ к своему почтовому ящику и можете направлять письма по любым известным Вам адресам.

После регистрации в почтовом ящике можно создать электронную визитную карточку, в которой указываются личные данные (это условие приемлимо, если только Вы хотите использовать свой ящик не в корыстных целях)

- фамилия и имя, а также координаты связи - адрес электронной почты телефон и личный сайт, если он есть.

После этого письмами с визиткой сообщите адрес своего почтового ящика своим знакомым, имеющим электронную почту. Через некоторое время к Вам начнут поступать электронные письма, которые будут накапливаться в Вашем почтовом ящике.

В электронных почтовых ящиках можно создать набор папок.

10.4.4. Телеконференции

Новости – это одно из старейших в истории Интернета средств коммуникации между группами людей, интересующимися одним определенным вопросом. Новости Usenet (от англ. user 's network, сеть пользователей) изобретены тремя американскими студентами в 1979 году. Usenet служила в то время для распространения информации и новостей по программированию . Данные сортировались по пятнадцати рубрикам , впоследствии получившим название " группы новостей " , " конференции " или "*телеконференции*".

Телеконференция – это электронная газета, состоящая целиком из объявлений ее подписчиков (электронная доска объявлений).

Для удобства телеконференции разбиты по темам, любой абонент сети может участвовать в понравившихся телеконференциях – подписаться на них, получать из них материалы и отправлять туда свои объявления.

Телеконференции могут быть коммерческими и некоммерческими, последние – платными и бесплатными. Обычно в каждой телеконференции существует свой устав, определяющий ее тематику и правила ее использования.

Если обыкновенную электронную почту можно сравнить с частной перепиской между двумя корреспондентами , то новости больше похожи на газетную публикацию . Процесс распространения новостей выглядит примерно так : человек , желающий опубликовать сообщение , посылает письмо специального формата на сервер новостей (специальный компьютер) .Это сообщение обрабатывается и начинает распространяться между всеми

остальными серверами , подписанными на данную тему (группу новостей) , или , как их еще называют , телеконференций . Сервер новостей мнаходит своего ближайшего соседа и передает ему накопившиеся новости. Тот , в свою очередь , передает их следующему соседу , и процесс продолжается , в результате чего однажды посланное письмо всего через несколько часов оказывается многократно размноженным и разлетается буквально по всему свету . Каждый человек , подписанный на определенную конференцию , в результате ознакомится и с вашим посланием .

В настоящее время количество информации, передаваемое при помощи серверов новостей, измеряется гигабайтами в сутки и продолжает стабильно увеличиваться. Чтобы обработать весь ежедневный поток новостей , требуются мощные серверы с дисками размером в десятки гигабайт и с быстрым соединением с Интернетом. Количество групп телеконференций по всему достигает десяти тысяч, а диапазон затрагиваемых в них тем порважает воображение: обсуждается все, начиная со структурных схем микропроцессоров фирмы Motorola и заканчивсая секрктами любителей уничтожать домашних животных у соседей. Между этими экзотическими темами лежит практически весь диапазон человеческой деятельности: секс, сетевые технологии, разговоры одиноких мужчин, обсуждение последних сериалов, прогрммирование на языке С , бизнес, финансы, вездесущая реклама и т.д.

Подписка на телеконференции.

Самой приятной особенностью подписки является ее полная автоматизация - обслуживанием подписчиков круглосуточно занимается специальная программа (сервер новостей - news-server). В любой время суток можно подписаться на понравившуюся тему или снять подписку.

Поскольку во многих конференциях (особенно коммерческих) объемы приходящих материалов чрезвычайно велики, то можно подписаться только на заголовки объявлений (что гораздо дешевле), а затем уже по ним заказывать заинтересовавшие статьи.

Основные семейства телеконференций.

Телеконференции могут быть *межсетевыми* и *внутрисетевыми*. *Внутрисетевая конференция* доступна только абонентам одной сети, *межсетевая* является общей для нескольких сетей. На больших узлах обычно имеются локальные конференции, часто они бывают доступны не только абонентам данного узла, но и другим абонентам сети.

С другой стороны, имеются международные межсетевые конференции, общие для абонентов нескольких глобальных сетей. Они, как правило, ведутся на английском языке.

Организация новостей

Для облегчения поиска интересующей вас группы все новости разделены на тематики и организованы иерархически, то есть сходные вопросы находятся в сисках рядом друг с другом. Такая структура напоминает структуру каталогов вашего жесткого диска. Программирование на языке С , например, обсуждается в группе новостей *comp.lang.c* (здесь *comp.lang* - иерархия, а *c* - название конференции) .Программирование на Pascal находится рядом, в той же иерархии, но группа, как нетрудно догадаться, называется pascal. Рисованные картинки публикуются в иерархии *alt.binaries.clip-art* (*alt.binaries* - название иерархии, *clip-art* - наименование группы). С творчеством поэтов и писателей на русском языке можно познакомиться в телеконференции *relcom.arts.qwerty*, а с курсом доллара в конференции с характерным названием *relcom.currency*.

Чтение и отправка новостей

Как Netscape Navigator, так и Internet News предоставляет примерно одинаковые возможности для чтения , составления и отправки новостей . Если программа для работы с новостями от Micrisoft запускается из меню программ отдельно от самого броузера Internet Explorer , то программа Netscape News тесно связана с броузером Netscape Navigator и запускается из его меню.

Описывая настройку броузера на работу с новостями, мы предполагаем, что он уже настроен на чтение и отправку эдектронной почты, то есть информация о вашем почтовом адресе, имени и т.д. уже внесена в соответствующие поля настройки.

10.4.5. Получение файлов по FTP

HTML – формат, рассмотренный нами в предыдущих главах, в основном предназначен для передачи текстовой информации. Но в Интернете существует громадное количество файлов (в основном - программы вместе с документацией и т.п.). Размещать их в WEB было бы неразумно, так как все, что требуется от Сети в этом случае, заключается в простой пересылке данных с компьютера - архива на компьютер пользователя. Такие компьютеры - архивы называются *ftp-серверами*. Сокращение ftp обозначает File Transfer Protocol , протокол передачи файлов. Сам ftp и средства доступа по ftp появились гораздо раньше Web - браузеров и языка HTML. И это не удивительно, так как передача двоичных данных с компьютера на компьютер всегда была главнейшей задачей Интернета.

Анонимные FTP - серверы

Большинствол ftp-серверов, с которыми вам придется столкнуться, являются анонимными, так как вход туда открыт почти все время любому желающему без существенных ограничений. В мире существует огромное количество ftp - серверов: от крупнейших, хранящих сотни гигабайт информации (http://sunsite .unc.edu), до небольших, посвященных определенно теме. Таковым, например, может оказаться ftp - сервер вашего провайдера, содержащий программы, необходимые для работы клиентов.

Для входа в анонимный ftp - сервер в качестве имени пользователя следует указать волшебное слово "anonymous ". Пароль можно ввести любой, но принято вводить свой адрес электронной почты.

Поиск файлов в FTP - архивах(archie)

Для облегчения поиска была придумана специальная система *Archie*, знающая почти о каждом ftp - архиве, зарегестрированном в ее базе данных .*Archie* состоит из двух частей: клиента и сервера. В мире существует несколько *Archie* - серверов, регулярно опрашивающих ftp - архивы и выясняющих, какие файлы где находятся. Все, что нужно для доступа к этой базе данных - простая программа *Archie* - клиент. Запустившись, она связывается с заранее выбранным сервером и задает ему вопрос типа: " Скажи мне, на каком из известных тебе ftp - серверов находится файл с таким - то именем?" Если *Archie* - серверу известно местоположение файла, он выдает список ftp - серверов, с которых этот файл можно скачать. Если нет - увы , придется либо исправить ошибку в названии искомого файла, либо продолжать поиски вручную (или с помощью других поисковых инструментов). Обратите внимание на то, что сам по себе *Archie* не представляет для передачи файлов. Он лишь снабжает вкас указателем на требуемый ресурс. Некоторые программы *Archi* , правда, нарушает это правило; с их помощью можно не только найти файл, но и тут же скачать его по ftp. К числу таких клиентов относится FRArchie, с которым мы сейчас и познакомимся.

Тема 11. Модели решения функциональных и вычислительных задач – 2 часа

11.1. Виды моделей

Практически во всех науках о природе, живой и неживой, об обществе, построение и использование моделей является мощным орудием познания. Реальные объекты и процессы бывают столь многогранны и сложны, что лучшим способом их изучения часто является построение модели, отобража-

ющей лишь какую-то грань реальности и поэтому многократно более простой, чем эта реальность, и исследование вначале этой модели. Многовековой опыт развития науки доказал на практике плодотворность такого подхода.

В прикладных областях различают следующие виды абстрактных моделей:

1. *Вербальные (текстовые) модели.* Эти модели используют последовательности предложений на формализованных диалектах естественного языка для описания той или иной области действительности (примерами такого рода моделей являются милицейский протокол, правила дорожного движения).

2. *Математические модели* – очень широкий класс знаковых моделей (основанных на формальных языках над конечными алфавитами), широко использующих те или иные математические методы. Например, можно рассмотреть математическую модель звезды. Эта модель будет представлять собой сложную систему уравнений, описывающих физические процессы, происходящие в недрах звезды. Математической моделью другого рода являются, например, математические соотношения, позволяющие рассчитать оптимальный (наилучший с экономической точки зрения) план работы какого-либо предприятия.

3. *Информационные модели* – класс знаковых моделей, описывающих информационные процессы (возникновение, передачу, преобразование и использование информации) в системах самой разнообразной природы.

Граница между вербальными, математическими и информационными моделями может быть проведена весьма условно; вполне возможно считать информационные модели подклассом математических моделей. Однако, в рамках информатики как самостоятельной науки, отделенной от математики, физики, лингвистики и других наук, выделение информационных моделей в отдельный класс является целесообразным.

Отметим, что существуют и иные подходы к классификации абстрактных моделей; общепринятая точка зрения здесь еще не установилась. В частности, есть тенденция резкого расширения содержания понятия "информационная модель", при котором информационное моделирование включает в себя и вербальные, и математические модели.

11.2. Математическое моделирование

Математическая модель выражает существенные черты объекта или процесса языком уравнений и других математических средств. Собственно говоря, сама математика обязана своим существованием тому, что она пытается отразить, т.е. промоделировать, на своем специфическом языке закономерности окружающего мира.

Путь математического моделирования в наше время гораздо более всеобъемлющ, нежели моделирования натурного. Огромный толчок развитию математического моделирования дало появление ЭВМ, хотя сам метод зародился одновременно с математикой тысячи лет назад.

Математическое моделирование как таковое отнюдь не всегда требует компьютерной поддержки. Каждый специалист, профессионально занимающийся математическим моделированием, делает все возможное для аналитического исследования модели. *Аналитические решения* (т.е. представленные формулами, выражающими результаты исследования через исходные данные) обычно удобнее и информативнее *численных*. Возможности аналитических методов решения сложных математических задач, однако, очень ограниченны и, как правило, эти методы гораздо сложнее численных. В данной главе доминируют численные методы, реализуемые на компьютерах. Это связано с тем, что моделирование здесь рассматривается под углом зрения компьютерных (информационных) технологий. Такой подход несколько сужает возможности метода в целом; его достоинство - некоторое снижение барьера необходимой математической подготовки (хотя, разумеется, и в численные методы при профессиональном занятии математическим моделирова-

нием приходится углубляться настолько, что при этом требуется значительное математическое образование). Наконец, отметим, что понятия *"аналитическое решение"* и *"компьютерное решение"* отнюдь не противостоят друг другу, так как:

а) все чаще компьютеры при математическом моделировании используются не только для численных расчетов, но и для аналитических преобразований;

б) результат аналитического исследования математической модели часто выражен столь сложной формулой, что при взгляде на нее не складывается восприятия описываемого ей процесса. Эту формулу (хорошо еще, если просто формулу!) нужно протабулировать, представить графически, проиллюстрировать в динамике, иногда даже озвучить, т.е. проделать то, что называется "визуализацией абстракций". При этом компьютер - незаменимое техническое средство.

11.3. Этапы и цели математического моделирования

Здесь мы рассмотрим процесс компьютерного математического моделирования, включающий численный эксперимент с моделью.

Первый этап – определение целей моделирования. Основные из них таковы:

•модель нужна для того, чтобы понять, как устроен конкретный объект, какова его структура, основные свойства, законы развития и взаимодействия с окружающим миром **(***понимание***);**

•модель нужна для того, чтобы научиться управлять объектом (или процессом) и определить наилучшие способы управления при заданных целях и критериях **(***управление***);**

•модель нужна для того, чтобы прогнозировать прямые и косвенные последствия реализации заданных способов и форм воздействия на объект **(***прогнозирование***)**.

Важнейшим этапом моделирования является разделение входных параметров по степени важности влияния их изменений на выходные. Такой процесс называется *ранжированием* (разделением по рангам).

Следующий этап – поиск математического описания. На этом этапе необходимо перейти от абстрактной формулировки модели к формулировке, имеющей конкретное математическое наполнение. В этот момент модель предстает перед нами в виде уравнения, системы уравнений, системы неравенств, дифференциального уравнения или системы таких уравнений и т.д. На основе поиска математического описания строится математическая модель.

11.4. Классификация математических моделей

Выделяют следующие математические модели:

1) *дескриптивные* (описательные) модели;

2) *оптимизационные* модели;

3) *многокритериальные* модели;

4) *игровые* модели;

5) *имитационные* модели.

Остановимся на этом чуть подробнее и поясним на примерах. Моделируя движение кометы, вторгшейся в Солнечную систему, мы описываем (предсказываем) траекторию ее полета, расстояние, на котором она пройдет от Земли и т.д., т.е. ставим чисто *описательные* цели. У нас нет никаких возможностей повлиять на движение кометы, что-то изменить.

На другом уровне процессов мы можем воздействовать на них, пытаясь добиться какой-то цели. В этом случае в модель входит один или несколько параметров, доступных нашему влиянию. Например, меняя тепловой режимов зернохранилище, мы можем стремиться подобрать такой, чтобы достичь максимальной сохранности зерна, т.е. *оптимизируем* процесс.

Часто приходится оптимизировать процесс по нескольким параметрам сразу, причем цели могут быть весьма противоречивыми. Например, зная

цены на продукты и потребность человека в пище, организовать питание больших групп людей (в армии, летнем лагере и др.) как можно полезнее и как можно дешевле. Ясно, что эти цели, вообще говоря, совсем не совпадают, т.е. при моделировании будет *несколько критериев*, между которыми надо искать баланс.

Игровые модели могут иметь отношение не только к детским играм (в том числе и компьютерным), но и к вещам серьезным. Например, полководец перед сражением в условиях наличия неполной информации о противостоящей армии должен разработать план, в каком порядке вводить в бой те или иные части и т.д., учитывая и возможную реакцию противника. Есть специальный достаточно сложный раздел современной математики - *теория игр*, - изучающий методы принятия решений в условиях неполной информации.

Наконец, бывает, что модель в большой мере подражает реальному процессу, т.е. *имитирует* его. Например, моделируя изменение (динамику) численности микроорганизмов в колонии, можно рассматривать много отдельных объектов и следить за судьбой каждого из них, ставя определенные условия для его выживания, размножения и т.д. При этом иногда явное математическое описание процесса не используется, заменяясь некоторыми словесными условиями (например, по истечении некоторого отрезка времени микроорганизм делится на две части, а другого отрезка - погибает). Другой пример - моделирование движения молекул в газе, когда каждая молекула представляется в виде шарика, и задаются условия поведения этих шариков при столкновении друг с другом и со стенками (например, абсолютно упругий удар); при этом не нужно использовать никаких уравнений движения. Можно сказать, что чаще всего *имитационное моделирование* применяется в попытке описать свойства большой системы при условии, что поведение составляющих ее объектов очень просто и четко сформулировано. Математическое описание тогда производится на уровне статистической обработки результатов моделирования при нахождении макроскопических характеристик системы. Такой компьютерный эксперимент фактически претендует на вос-

произведение натурного эксперимента; на вопрос "зачем же это делать?" можно дать следующий ответ: имитационное моделирование позволяет выделить "в чистом виде" следствия гипотез, заложенных в наши представления о микрособытиях, очистив их от неизбежного в натурном эксперименте влияния других факторов, о которых мы можем даже не подозревать. Если же, как это иногда бывает, такое моделирование включает и элементы математического описания событий на микроуровне, и если исследователь при этом не ставит задачу поиска стратегии регулирования результатов (например, управления численностью колонии микроорганизмов), то отличие имитационной модели от дескриптивной достаточно условно; это, скорее, вопрос терминологии.

11.5. Моделирование случайных процессов

Понятие "*случайный*" – одно из самых фундаментальных как в математике, так и в повседневной жизни. Моделирование случайных процессов - мощнейшее направление в современном математическом моделировании.

Событие называется *случайным*, если оно достоверно непредсказуемо. Случайность окружает наш мир и чаще всего играет отрицательную роль в нашей жизни. Однако есть обстоятельства, в которых случайность может оказаться полезной.

В сложных вычислениях, когда искомый результат зависит от результатов многих факторов, моделей и измерений, можно сократить объем вычислений за счет случайных значений значащих цифр. Из теории эволюции следует, что случайность проявляет себя как конструктивный, позитивный фактор. В частности, естественный отбор реализует как бы метод проб и ошибок, отбирая в процессе развития особи с наиболее целесообразными свойствами организма. Далее случайность проявляется в множественности ее результатов, обеспечивая гибкость реакции популяции на изменения внешней среды.

В силу сказанного имеет смысл положить случайность в основу методов получения решения посредством проб и ошибок, путем случайного поиска.

Метод статистического моделирования имеет множество приложений. Чаще всего он заключается в том, что для решения математической задачи строится некоторая случайная величина z, такая , что математическое ожидание этой случайной величины E(z) является значением искомого решения. Проводя достаточное количество раз эксперимент со случайной величиной z, можно найти приближенное решение как среднее значение результатов эксперимента.

Общеизвестно, сколь важно для решения экономических задач планирование - как при рыночной, так и при плановой экономике. Обычно для решения экономической проблемы существует много способов (стратегий), отнюдь не равноценных по затратам финансов, людских ресурсов, времени исполнения, а также по достигаемым результатам. Наилучший из способов (по отношению к выбранному критерию - одному или нескольким) называют *оптимальным***.**

Тема 12. Компьютерная графика и системы геометрического моделирования – 2 часа

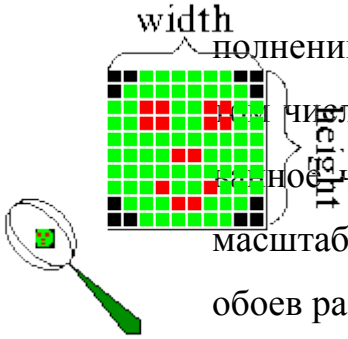
Представление данных на мониторе компьютера в графическом виде впервые было реализовано в середине 50-х годов для больших ЭВМ, применявшихся в научных и военных исследованиях. С тех пор графический способ отображения данных сталь неотъемлимой принадлежностью компьютерных систем.

Существует специальная область информатики, изучающая методы и средства создания и обработки изображений с помощью программно-аппаратных вычислительных комплексов – *компьютерная графика*.

В зависимости от принципа вывода изображения на экран монитора различают *растровую, векторную* и *фрактальную графику* и соответствующие графические редакторы. Отдельным видом компьютерной графики является *трехмерная графика*, сочетающая векторный и растровый способ построения изображения.

Графические редакторы используют для создания и обработки изображений. Главные функции редактора:

1. создание графических изображений;
2. их редактирование.



Под редактированием понимают ввод изменений, исправлений и дополнений. Редактировать можно созданные изображения, а также готовые, в том числе и сканированные. Можно редактировать и изображение, скопированное через буфер обмена из другого приложения. Изображения можно масштабировать, вращать, растягивать. Их также можно сохранять в виде обоев рабочего стола.

12.1. Растровая графика

Растровый формат, с которым мы познакомимся подробнее, характеризуется тем, что все изображение по вертикали и горизонтали разбивается на достаточно мелкие прямоугольники -- так называемые элементы изображения, или *пикселы* (от английского ***pixel*** *-- picture element*).

В файле, содержащем растровую графику, хранится информация о цвете каждого пиксела данного изображения. Чем меньше прямоугольники, на которые разбивается изображение, тем больше разрешение ***(resolution)***, то есть, тем более мелкие детали можно закодировать в таком графическом файле.

Размер ***(size)*** изображения, хранящегося в файле, задается в виде числа пикселов по горизонтали ***(width)*** и вертикали ***(height)***. Для примера, оптимальное разрешение 15-дюймового монитора, как правило, составляет 1024x768.

12.1.1. Глубина цвета

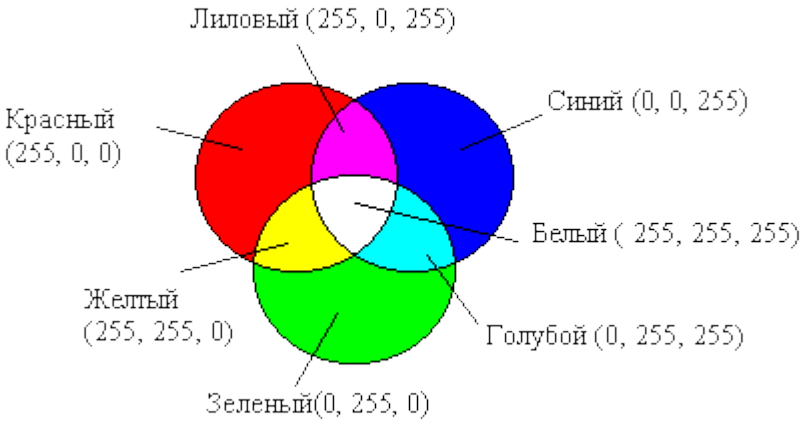
Кроме размера изображения, важной является информация о количестве цветов, закодированных в файле. Цвет каждого пиксела кодируется определенным числом бит ***(bit)***, то есть элементарных единиц информации, с которыми может иметь дело компьютер. Каждый бит может принимать два значения -- 1 или 0. В зависимости от того, сколько бит отведено для цвета каждого пиксела, возможно кодирование различного числа цветов. Нетрудно сообразить, что если для кодировки отвести лишь один бит, то каждый пиксел может быть либо белым (значение 1), либо черным (значение 0). Такое изображение называют монохромным ***(monochrome)***.

Далее, если для кодировки отвести четыре бита, то можно закодировать 2^4=16 различных цветов, отвечающих комбинациям бит от 0000 до 1111. Если отвести 8 бит -- то такой рисунок может содержать 2^8=256 различных цветов (от 00000000 до 11111111), 16 бит -- 2^{16}=65 536 различных цветов (так называемый ***High Color***). И, наконец, если отвести 24 бита, то потенциально рисунок может содержать 2^{24}=16 777 216 различных цветов и оттенков -- вполне достаточно даже для самого взыскательного художника! В последнем случае кодировка называется *24-bit* ***True Color***. Следует обратить внимание на слово "потенциально": даже если в файле и отводится 24 бита на каждый пиксел, это еще не означает, что вы действительно сможете насладиться та-

кой богатой палитрой - ведь технические возможности мониторов ограничены.

12.1.2. RGB-модель

Способ разделения цвета на составляющие компоненты называется **Цветовой моделью**. В компьютерной графике применяются три цветовые модели: **RGB**, **CMYK** и **HSB**.



Наиболее распространенным способом кодирования цвета является модель **RGB**. При этом способе кодирования любой цвет представляется в виде комбинации трех цветов: красного ***(Red)***, зеленого ***(Green)*** и синего ***(Blue)***, взятых с разной интенсивностью. Интенсивность каждого из трех цветов -- это один байт (т. е. число в диапазоне от 0 до 255), который хорошо представляется двумя 16-ричными цифрами (числом от 00 до FF). Таким образом, цвет удобно записывать тремя парами 16-ричных цифр, как это принято, например, в HTML-документах.

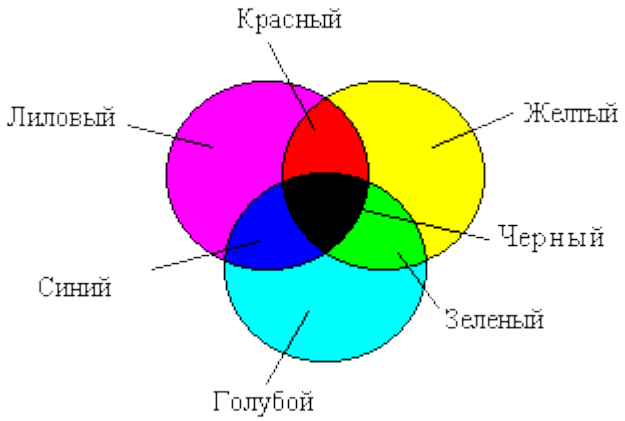
Пример.

В языке гипертекстовой разметки документов HTML цвета можно задавать так: черный -- 000000, белый -- FFFFFF, желтый -- FFFF00 и т. д.; чтобы получить более темный желтый цвет, надо одинаково уменьшить интенсивности красного и зеленого -- A7A700.

Чем больше значение байта цветовой составляющей, тем ярче этот цвет. При наложении одной составляющей на другую яркость суммарного цвета также увеличивается.

12.1.3. Цветовая модель CMYK

Цветовая модель **CMYK** соответствует рисованию красками на бумажном листе и используется при работе с отраженным цветом, т. е. для подготовки печатных документов.



Цветовыми составляющими этой модели являются цвета: голубой *(Cyan)*, лиловый *(Magenta)*, желтый *(Yellow)* и черный *(Black)*. Эти цвета получаются в результате вычитания основных цветов модели **RGB** из белого цвета. Черный цвет задается отдельно. Увеличение количества краски приводит к уменьшению яркости цвета.

12.1.4. Цветовая модель HSB

Системы цветов **RGB** и **CMYK** связаны с ограничениями, накладываемыми аппаратным обеспечением (монитор компьютера в случае **RGB** и типографские краски в случае **CMYK**).

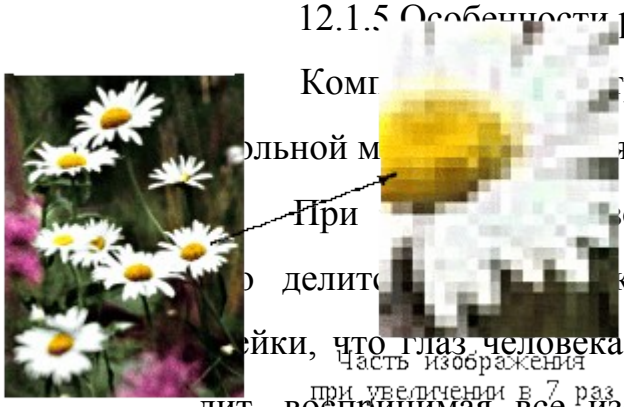
Цветовая модель **HSB** наиболее удобна для человека, т. к. она хорошо согласуется с моделью восприятия цвета человеком. Компонентами модели **HSB** являются:

- тон *(Hue)*;
- насыщенность *(Saturation)*;
- яркость цвета *(Brightness)*.

Тон - это конкретный оттенок цвета. Насыщенность характеризует его интенсивность или чистоту. Яркость же зависит от примеси черной краски, добавленной к данному цвету.

Значение цвета выбирается как вектор, выходящий из центра окружности. Точка в центре соответствует белому цвету, а точки по границе окружности - чистым цветам. Направление вектора определяет цветовой оттенок и задается в угловых градусах. Длина вектора определяет насыщенность цвета. Яркость цвета задают на отдельной оси.

12.1.5 Особенности растровой графики



Комп[illegible]ровое изображение представляется в виде прямо[illegible]льной м[illegible]я ячейка которой представлена цветной точкой.

При [illegible]ображения [illegible]о делит[illegible]крошечные [illegible]ейки, что глаз человека их не видит, воспринимая все изображение как целое. Сама сетка получила название *растровой карты*, а ее единичный элемент называется *пикселом.*

Пикселы подобны зернам фотографии и при значительном увеличении они становятся заметными. Растровая карта представляет собой набор (массив) троек чисел: две координаты пиксела на плоскости и его цвет.

В отличие от векторных изображений, при создании объектов растровой графики математические формулы не используются, поэтому для синтеза растровых изображений необходимо задавать разрешение и размеры изображения.

С помощью растровой графики можно отразить и передать всю гамму оттенков и тонких эффектов, присущих реальному изображению. Растровое изображение ближе к фотографии, оно позволяет более точно воспроизводить основные характеристики фотографии: освещенность, прозрачность и глубину резкости.

Чаще всего растровые изображения получают с помощью сканирования фотографий и других изображений, с помощью цифровой фотокамеры или путем "захвата" кадра видеосъемки. Растровые изображения можно получить и непосредственно в программах растровой или векторной графики путем преобразовании векторных изображений.

Существует множество форматов файлов растровой графики, и каждый из них предусматривает собственный способ кодирования информации об изображении. Перечислим особенности лишь наиболее распространенных форматов.

Формат	Макс. число бит/пиксел	Макс. число цветов	Макс. размер изображения, пиксел	Методы сжатия	Кодирование нескольких изображений
BMP	24	16 777 216	65535 x 65535	RLE	-
GIF	8	256	65535 x 65535	LZW	+
JPEG	24	16 777 216	65535 x 65535	JPEG	-
PCX	24	16 777 216	65535 x 65535	RLE	-
PNG	48	281 474 976 710 656	2 147 483 647 x 2 147 483 647	Deflation (вариант LZ77)	-
TIFF	24	16 777 216	всего 4 294 967 295	LZW, RLE и другие	+

Из большого числа форматов графических файлов в Интернете сейчас широко используются только два -- GIF и JPEG. О них и поговорим подробнее.

12.1.6. GIF – формат

Популярный формат GIF разработан фирмой *CompuServe*, как не зависящий от аппаратного обеспечения. Он предназначен для хранения растровых изображений с сжатием. В одном файле этого формата может храниться несколько изображений. Обычно эта возможность используется для хранения анимированных изображений (как набор кадров).

GIF-формат позволяет записывать изображение "через строчку" (Interlaced), благодаря чему, имея только часть файла, можно увидеть изобра-

жение целиком, но с меньшим разрешением. Эта возможность широко применяется в Интернет. Сначала вы видите картинку с грубым разрешением, а по мере поступления новых данных ее качество улучшается. Основное ограничение формата GIF состоит в том, что цветное изображение может содержать не более 256 цветов. Для полиграфии этого явно недостаточно.

12.1.7. JPEG – формат

Формат файла JPEG (*Joint Photographic Experts Group -- Объединенная экспертная группа по фотографии*) был разработан компанией *C-Cube Microsystems*, как эффективный метод хранения изображений с большой глубиной цвета, например, получаемых при сканировании фотографий с многочисленными едва уловимыми (а иногда и неуловимыми) оттенками цвета.

Самое большое отличие формата JPEG от других форматов состоит в том, что в JPEG используется **алгоритм сжатия с потерями** (а не алгоритм без потерь).

Алгоритм сжатия без потерь так сохраняет информацию об изображении, что распакованное изображение в точности соответствует оригиналу. При сжатии с потерями приносится в жертву часть информации об изображении, чтобы достичь большего коэффициента сжатия.

Сжатие, используемое в формате JPEG, необратимо искажает изображение. Это не заметно при его простом просмотре, но становится явным при последующих манипуляциях. Зато размер файла получается от 10 до 500 раз меньше, чем BMP! Если вы решили записать изображение в формате JPEG, то лучше выполнить все необходимые операции перед первой записью файла.

Сравнение GIF и JPEG

1. GIF-формат удобен при работе с рисованными картинками.
2. JPEG-формат лучше использовать для хранения фотографий и изображений с большим количеством цветов.

3. Для создания анимации и изображений с прозрачным фоном применяется GIF-формат.

12.2. Векторная графика

Основным логическим элементом векторной графики является геометрический объект. В качестве объекта принимаются простые геометрические фигуры (так называемые примитивы -- прямоугольник, окружность, эллипс, линия), составные фигуры или фигуры, построенные из примитивов, цветовые заливки, в том числе градиенты.

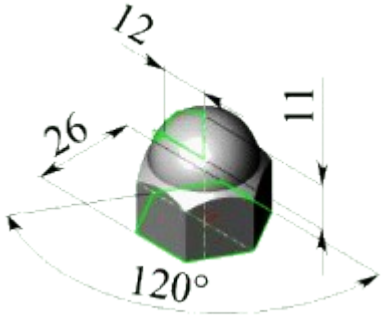
Преимущество векторной графики заключается в том, что форму, цвет и пространственное положение составляющих ее объектов можно описывать с помощью математических формул.

Важным объектом векторной графики является сплайн. Сплайн - это кривая, посредством которой описывается та или иная геометрическая фигура. На сплайнах построены современные шрифты *TryeType* и *PostScript*.

У векторной графики много достоинств. Она экономна в плане дискового пространства, необходимого для хранения изображений: это связано с тем, что сохраняется не само изображение, а только некоторые основные данные, используя которые, программа всякий раз воссоздает изображение заново. Кроме того, описание цветовых характеристик почти не увеличивает размер файла.

Объекты векторной графики легко трансформируются и модифицируются, что не оказывает практически никакого влияния на качество изображения. Масштабирование, поворот, искривление могут быть сведены к паретройке элементарных преобразований над векторами.

В тех областях графики, где важное значение имеет сохранение ясных и четких контуров, например, в шрифтовых композициях, в создании логотипов и прочее, векторные программы незаменимы.



Векторная графика может включать в себя и фрагменты растровой графики: фрагмент становится таким же объектом, как и все остальные (правда, со значительными ограничениями в обработке).

Важным преимуществом программ векторной графики является развитые средства интеграции изображений и текста, единый подход к ним. Поэтому программы векторной графики незаменимы в области дизайна, технического рисования, для чертежно-графических и оформительских работ.

Однако, с другой стороны, векторная графика может показаться чрезмерно жесткой, "фанерной". Она действительно ограничена в чисто живописных средствах: в программах векторной графики практически невозможно создавать фотореалистические изображения.

А кроме того, векторный принцип описания изображения не позволяет автоматизировать ввод графической информации, как это делает сканер для точечной графики.

В последнее время все большее распространение получают программы 3-мерного моделирования, также имеющие векторную природу.

Обладая изощренными методами отрисовки (метод трассировки лучей, метод излучательности), эти программы позволяют создавать фотореалистичные растровые изображения с произвольным разрешением из векторных объектов при умеренных затратах сил и времени.

В любом случае, если вы работаете с графикой, то неизбежно будете иметь дело с обеими ее формами - векторной и растровой. Понимание их

сильных и слабых сторон позволит вам выполнить свою работу максимально эффективно.

12.3. Средства работы с графикой

Существует множество различных графических редакторов. Некоторые из них, такие как Adobe PhotoShop и CorelDraw, предназначены для проффесиональной работы с графикой. Это коммерческие продукты, которые стоят немалых денег. Другие, например Paint, встроенный в ОС Windows, доступны для работы даже маленькому ребенку, но и возможности у них не велики.

В ОС Linux на данный момент представлен весь спектр графических редакторов. Первое место среди них, бесспорно занимает мощнейший редактор GIMP, обладающий инструментами как для создания десятков разновидностей форматов **растровой графики,** так и средствами подготовки **векторной графики.** В его состав входит уникальный редактор **фрактальной графики**, позволяющий создавать настоящие произведения искусства. Следует также отметить, что, как и большинство Linux-приложений, редактор GIMP распространяется бесплатно, что также является немаловажным плюсом в пользу его применения.

Имеются в ОС Linux и другие графические редакторы, более специализированные для работы с тем или иным форматом графики. Среди них:

- XPaint – редактор растровой графики, похожий на Paint, но содержащий массу дополнительных возможностей;
- KPaint – встроенный в KDE редактор растровой графики, менее функционален чем XPaint;
- KDE Icon Editor – редактор растровой графики, предназначенный для создания пиктограмм;
- XFig – редактор векторной графики.

12.3.1. Редактор Paint

Этот редактор является одним из самых простых графических редакторов в среде MS Windows. Он появился в версии Windows 95 и с тех пор не изменился. **Paint** (в переводе с английского означает *краска*) представляет собой простой однооконный редактор растровой графики, который, тем не менее, позволяет создать достаточно сложный рисунок.

Программа включает средства для построения прямых и кривых линий, эллипсов и окружностей, прямоугольников, квадратов и многоугольников (как контурных, так и закрашенных). Есть инструменты для выделения фрагмента рисунка, заливки замкнутой области цветом, а также инструменты, имитирующие рисование кистью и пульверизатором. Имеется возможность создания надписи и задания толщины линии.

Доступны и некоторые операции преобразования рисунка, а именно: зеркальное отображение относительно горизонтальной и вертикальной оси, инвертирование и замена цветов, сжатие, растяжение и наклон. Однако, в **Paint** совершенно отсутствуют разного рода эффекты и фильтры. Кроме того, этот редактор поддерживает всего несколько форматов файлов.

12.3.2. Редактор Adobe Photoshop

Боле серьезным инструментом является *редактор Adobe Photoshop*.

В редакторе *Adobe Photoshop* можно редактировать следующие характеристики изображения:

1. Характеристики (атрибуты) изображения можно определить и изменить в диалоговом окне **Размер Изображения**, которое появляется, если щёлкнуть в главном меню команду **Изображение | Размер изображения**.

2. Важной характеристикой изображения является **Режим** (*Mode*), он определяет глубину цвета, т.е. количество битов, которым кодируется каждый пиксел. Изображения, обрабатываемые компьютером, как и остальная информация, кодируются (оцифровываются). Режим можно определить и из-

менить в диалоговом **окне Новый**, которое появляется, если щёлкнуть в главном меню команду **Файл | Новый**.

Режимы бывают:

1. ***Bitmap*** (битовая карта) – каждый пиксел кодируется одним битом, которым можно передать только два состояния, два цвета: черный и белый.
2. ***Ч-Б*** – характеризуется значением яркости в интервале от 0 (чёрный цвет) до 256 (белый цвет), остальные значения для передачи полутонов – градаций серого.
3. ***RGB (True Color)*** – каждый пиксел кодируется 24 битами, что позволяет закодировать и передать 16,7 млн. оттенков. Каждый цвет формируется тремя составляющими (каналами): красный (Red), зелёный (Green), синий (Blue). Каждая составляющая – это 8 бит ($8 \times 3=24$).
4. ***CMYK*** – каждый цвет формируется тремя составляющими: голубой (Cyan), пурпурный (Magenta), желтый (Yellow), чёрный (blacK). Цвета комбинируются образуя все другие цвета.
5. ***Lab*** – в этом режиме цвет определяется освещённостью (L) и двумя цветными составляющими: a – изменяется в диапазоне от зелёного до красного, b – изменяется в диапазоне от синего до жёлтого.

Тема 13. Интегрированные автоматизированные системы – 2 часа

13.1.Системы реального времени

Основная задача *системы реального масштаба времени (RT)* – получение правильных результатов за определенный крайний срок. Следовательно, вычислительная правильность системы зависит от двух составляющих: логической правильности результатов, и правильности выбора времени, то есть способности выполнения вычислений за крайние сроки. О *жестких системах реального масштабе времени (HRT)* можно думать как о специфическом подклассе RT систем, в котором неспособность удовлетворять вышеупомянутым крайним срокам может окончиться катастрофическим отказом системы.

В дальнейшем мы будем использовать фразу *"мягкая система реального масштаба времени (SRTT)"* для определения тех RT систем, в которых способность встречать крайние сроки действительно требуется; однако, отказ выполнения не приводит к отказу системы. Для проектов HRT и SRT систем могут встретиться такие сложности, как выбор времени применения и требования к ресурсам, и пригодность ресурсов системы. В частности в проекте HRT системы, которая поддерживает критические процессы (например, система управления полетами, система управления атомной электростанции, железнодорожные системы управления), эта сложность может быть усилена такими противоречивыми прикладными требованиями, как требованием на высоко надежные и высоко доступные услуги, и потребностью обеспечить эти услуги при удовлетворении строгих ограничений выбора времени. В частности, поскольку HRT система должна обеспечить услуги, которые, должны быть и своевременными и высоко доступными, проект любой такой системы требует, чтобы соответствующие методы разрешения ошибок, способные к встрече с жесткими требованиями в реальном масштабе времени, были развернуты в пределах этой системы. Текущая технология позволяет проектировщику системы HRT осуществлять рентабельные методы разрешения ошибок, основанные на использовании избыточных компонентов системы. Однако, развитию политики управления избыточности, которая выполняет требования в реальном масштабе времени, может препятствовать дальнейшее усложнение проекта системы. Таким образом, в сущности, проект HRT системы нуждается в тщательной оценке требований выполнения / надежности. И HRT и SRT системы могут быть построены из географически рассеянных ресурсов, связанных некоторой сетью, чтобы формировать распределенную RT систему (распределенные HRT системы могут классифицироваться как отзывчивые системы, то есть распределеные, ошибко-устойчивые, в реальном масштабе времени).

13.1.1. Проблемы разработки проекта

Любая система реального масштаба времени может быть описана как состоящая из *трех основных подсистем*. *Управляемая подсистема* (например, индустриальный завод, управляемое компьютером транспортное средство), диктует требования в реальном масштабе времени; *подсистема контроля* управляет некоторыми вычислениями и связью с оборудованием для использования от управляемой подсистемы; *подсистема оператора* контролирует полную деятельность системы. Интерфейс между управляемыми и подсистемами контроля состоит из таких устройств как датчики и приводы. Интерфейс между управляющей подсистемой, и оператором связывает человека с машиной. Управляемая подсистема представлена задачами (в дальнейшем называемыми прикладными задачами) которые используют оборудование, управляемое подсистемой контроля. Эта последняя подсистема может быть построена из очень большого количества процессоров, управляющими такими местными ресурсами, как память и устройства хранения, и доступ к локальной сети в реальном масштабе времени. Эти процессоры и ресурсы управляются системой программного обеспечения, так мы называем *операционную систему реального масштаба времени (RTOS).* Развертывание RTOS в критической окружающей среде (например, руководство и навигационные системы) налагает серьезные требования надежности на проект и функционирование этих RTOS . Эти требования могут быть определены из соображений максимальной приемлемой вероятности отказа системы. Таким образом, например, системы управления полетом, типа используемого в Аэробусе А-320, требуют вероятности отказа в полете 10^(-10) за час. В системах управления транспортными средствами, в которых стоимость отказа может быть определенный количество в терминах экономического штрафа (например, системы для спутникового руководства, беспилотные подводные навигационные системы), требуют вероятности отказа порядка 10^(-6) - 10^(-7) в час. Методы разрешения ошибок, основанные на управлении избыточными аппаратными средствами ЭВМ и компонентов системы программного

обеспечения, обычно используются, чтобы выполнить эти требования надежности. Однако, выполнение этих методов, действительно определяющих надежность системы, требуют, чтобы частью ресурсов системы были отданы для обеспечения надежности

RT приложения

RT приложение может быть оформлено как набор сотрудничающих задач. Эти задачи могут классифицироваться, согласно их требованиям выбора времени, как задачи жесткие в режиме реального времени (HRT), мягкие в режиме реального времени (SRT), и не в режиме реального времени (NRT). HRT задача - задача, чье своевременное (и логически правильное) выполнение считают критическим для действия полной системы. Предельный срок, HRT задачи традиционно называют жестким крайним сроком, вследствие критического характера этой задачи. Неспособность системы удовлетворить жесткому крайнему сроку может окончиться крахом системы. SRT задача, так же, характеризуется крайним сроком, удовлетворение которому действительно желательна, хотя не критично, для функционирования систе мы (таким образом, крайний срок SRT задачи - обычно называемый мягким крайним сроком). NRT задачи - те задачи, которые не показывают никакие требования в реальном масштабе времени (например, задачи обслуживания системы, которые могут иногда выполняться в "фоне"). Прикладные задачи могут далее классифицироваться как периодические, апериодические (или асинхронные), и спорадические задачи. Периодические задачи - те задачи, которые входят в состояние выполнения периодически. Эти задачи, обычно используются в таких задачах как обработка сигнала и контроль , типично характеризуются жестким крайним сроком. Апериодические задачи - те задачи, чье выполнение, не может ожидаться, так как их выполнение, определено возникновением некоторых внутренних или внешних факторов (например, задача, отвечающая на запрос от оператора). Эти задачи обычно характеризуются мягким крайним сроком. Наконец, апериодические задачи, характеризо-

ванные к жестким крайним срокам, названы спорадическими задачами (например, задачи в критическом положении запрашивающие некоторые действия от оператора). RTOS должна гарантировать, что каждое индивидуальное выполнение каждой прикладной задачи выполняет требования выбора времени этой задачи. Однако, для выполнения этого условия нельзя поступиться временем среднего отклика каждой прикладной задачи. Фундаментальное свойство RTOS - системы предсказуемость, то есть функциональное и рассчитывающее поведение RTOS должно быть так детерминированный по мере необходимости, чтобы выполнить эту RTOS спецификация. Таким образом, быстрые аппаратные средства ЭВМ и эффективные алгоритмы действительно полезны, чтобы строить RTOS, которые выполняют требования в реальном масштабе времени; однако, они не достаточны для гарантирования предсказуемого поведения, требуемого от системы.

13.1.2. Планировщик заданий

В системах реального времени на алгоритм планировки возлагается задача определения последовательности выполнения заданий в соответствии с их требованиями к ресурсам и ко времени исполнения. При проектировании системы реального времени выбор приемлемого алгоритма планирования может зависеть от нескольких факторов: количества процессоров в системе, их гомогенности или гетерогенности, отношения предшествования между заданиями, метода синхронизации заданий. Кроме того, программно-зависимые характеристики заданий могут определять выбор алгоритма планирования. Например, задания могут быть прерываемыми и не прерываемыми. Выполнение прерываемого задания может быть прервано другим и продолжено позднее; непрерываемые задания выполняются до завершения и не могут быть прерваны. Приложению предлагаются обе возможности. (В этой работе мы не будем рассматривать непрерываемое планирование.) *Алгоритмы планирования* заданий могут быть разделены на *статические* и *динамические*. Статические алгоритмы определяют приемлемый план выполнения заданий

по их априорным характеристикам, динамический алгоритм модифицирует план во время исполнения заданий. Издержки на статическое планирование низки, но оно крайне нечувствительно и требует полной предсказуемости той системы реального времени, на которой оно установлено. Динамическое планирование связано с большими издержками, но способно адаптироваться к меняющемуся окружению.

В заключение приведем *основные особенности* систем реального времени в целом и управляющих систем реального времени в частности. Начнем с того, что "времязависимость" есть непременный атрибут системы реального времени. Но выполнение этого требования еще недостаточно для эффективной работы системы, так как в первую очередь система реального времени должна быть "предсказуемой". Мы описали две архитектуры, обеспечивающие предсказуемость системы: либо все действия в системе синхронизированы по времени, либо каждому действию предшествует определенное событие. Цель этих двух типов архитектур − удовлетворить требованию предсказуемости, применяя статическую или динамическую стратегии соответственно, чтобы выдержать требования к ресурсам и временные требования, предъявляемые к заданиям. Далее мы обсудили проблемы межпроцессного взаимодействия в системах реального времени. Были представлены принципиальные требования к коммуникационной инфраструктуре для поддержки приложений реального времени (а именно: задержка доступа к каналу связи, задержка сообщений связи и нерв задержки связи). Важнейшими характеристиками коммуникационного механизма, как мы выяснили в ходе обсуждения, являются: процент потери сообщений, скорость передачи сообщений, процент потери экстренных сообщений, эффективность использования канала передачи данных и масштабируемость механизма. Наконец, мы рассмотрели диспетчеризацию в системах реального времени и обсудили важнейшие влияющие факторы: обработка обращения к недоступным ресурсам, среднее время ответа на запрос, гарантированный процент вовремя обработанных запросов.

13.2. Автоматизированные Системы Научных Исследований (АСНИ)

Повышение эффективности фундаментальных и прикладных научных исследований становится важным фактором ускорения научно-технического прогресса.

Особое значение для повышения эффективности науки приобретает автоматизация научных исследований, позволяющая получать более точные и полные модели исследуемых объектов и явлений, ускорять ход научных исследований и снижать их трудоемкость, изучать сложные объекты и процессы, исследование которых традиционными методами затруднительно или невозможно. Применение *автоматизированных систем научных исследований* и комплексных испытаний образцов новой техники (АСНИ) наиболее эффективно в тех современных областях науки и техники, которые имеют дело с использованием больших объемов информации. К ним прежде всего относятся:

- ядерная физика (сбор и обработка экспериментальных данных, получаемых на реакторах, ускорителях и установках термоядерного синтеза);
- физика плазмы и твердого тела;
- радиофизика и электроника;
- астрономия и радиоастрономия;
- космические исследования (обработка информации, получаемой с искусственных спутников для нужд народного хозяйства);
- геология и геофизика (разведка полезных ископаемых);
- исследования Мирового океана, экологические исследования, прогнозирование погоды и стихийных бедствий;
- биология и медицина (исследования в области молекулярной биологии, микробиологического синтеза, диагностики заболеваний);
- химическая технология (моделирование технологических процессов, получение материалов с заданными свойствами);

- исследования сложных технологических процессов в промышленности;
- исследования и разработки в области энергетики (электростанции, сети электропередачи, энергетические системы);
- исследования и разработки в области транспортных коммуникаций, сетей связи и сетей вычислительных машин;
- натурные и стендовые испытания сложных технических объектов (летательных аппаратов, транспортных устройств, машин, сооружений);
- экономика, социальные исследования, право и языкознание.

Автоматизированные системы научных исследований и комплексных испытаний образцов новой техники обеспечивают получение значительного народнохозяйственного эффекта. Этот эффект образуется от повышения производительности труда в исследовательских и испытательных подразделениях, улучшения технико-экономических характеристик разрабатываемых объектов на основе получения и использования более точных моделей этих объектов, сокращения дорогостоящих натурных испытаний, исключения некоторых стадий опытно-конструкторских работ, что в конечном счете приводит к снижению затрат на разработку объектов новой техники.

АСНИ отличаются от других типов автоматизированных систем (АСУ, АСУТП, САПР и т.д.) характером информации, получаемой на выходе системы. Прежде всего, это обработанные или обобщенные экспериментальные данные, но главное - полученные на основе этих данных математические модели исследуемых объектов, явлений или процессов. Адекватность и точность таких моделей обеспечивается всем комплексом методических, программных и других средств системы. В АСНИ могут использоваться также и готовые математические модели для изучения поведения тех или иных объектов и процессов, а также для уточнения самих этих моделей. АСНИ поэтому являются системами для получения, корректировки или исследования моделей, используемых затем в других типах автоматизированных систем для управления, прогнозирования или проектирования.

Как правило, все типы АСНИ должны создаваться на базе серийных средств вычислительной техники широкого применения (процессоров, устройств памяти на магнитных лентах и дисках, печатающих устройств, дисплеев и т.п.). Однако, в АСНИ может примениться и специальная аппаратура для сопряжения ЭВМ с исследуемыми объектами. Эта аппаратура должна обеспечивать разнообразные функции предварительной обработки информации, иметь гибкую структуру и максимальную взаимозаменяемость модулей и блоков.

Поэтому создание аппаратуры сопряжения ЭВМ с объектами является одним из важнейших направлений работ, обеспечивающих эффективную разработку и развитие различных типов АСНИ. Блоки и модули аппаратуры сопряжения должны выпускаться серийно в соответствии с международными стандартами.

13.2.1. Цели создания АСНИ

АСНИ создаются в организациях и на предприятиях в целях:

обеспечения высоких темпов научно-технического прогресса;

повышения эффективности и качества научных исследований на основе получения или уточнения с помощью АСНИ математических моделей исследуемых объектов, явлений или процессов, а также применения этих моделей для проектирования, прогнозирования и управления;

повышения эффективности разрабатываемых с помощью АСНИ объектов, уменьшения затрат на их создание;

получения качественно новых научных результатов, достижение которых принципиально невозможно без применения АСНИ;

сокращения сроков, уменьшения трудоемкости научных исследований и комплексных испытаний образцов новой техники.

Достижение целей создания АСНИ обеспечивается путем:

систематизации и совершенствования процессов научных исследований и испытаний на основе применения математических методов и средств вычислительной техники;

комплексной автоматизации исследовательских работ в научно-исследовательской организации с необходимой перестройкой ее структуры и кадрового состава;

повышения качества управления научными исследованиями;

применения эффективных математических методов организации и планирования экспериментов;

использования методов обработки и представления результатов научных исследований и испытаний в виде математических моделей, имеющих заданную форму;

автоматизации трудоемких работ;

замены натурных испытаний и макетирования математическим моделированием.

13.2.2. Определение АСНИ

Автоматизированная система научных исследований и комплексных испытаний образцов новой техники (АСНИ) - это программно-аппаратный комплекс на базе средств вычислительной техники, предназначенный для проведения научных исследований или комплексных испытаний образцов новой техники на основе получения и использования моделей исследуемых объектов, явлений и процессов.

Программно-аппаратный комплекс АСНИ состоит из средств методического, программного, технического, информационного и организационно-правового обеспечения.

Взаимодействие исследуемого объекта, явления или процесса с АСНИ осуществляется через аппаратуру сопряжения, входящую в состав программно-аппаратного комплекса.

Взаимодействие подразделений научно-исследовательской организации или предприятия с АСНИ регламентируется средствами организационно-правового обеспечения системы.

13.2.3. Функции АСНИ

Основная функция АСНИ состоит в получении результатов научных исследований (комплексных испытаний) путем автоматизированной обработки экспериментальных данных и другой информации, получения и исследования моделей объектов, явлений и процессов на основе применения математических методов, автоматизированных процедур, планирования и управления экспериментом.

Автоматизированные процедуры в АСНИ состоят в том, что исследования (испытания) объектов, явлений и процессов, получение и исследование математических моделей осуществляется путем взаимодействия пользователя с АСНИ в режиме диалога.

В АСНИ могут осуществляться автоматические процедуры, при которых обработка данных, идентификация или построение математических моделей производятся без участия человека.

В АСНИ могут применяться также процедуры планирования и управления экспериментом, при которых использование моделирования корректирует условия эксперимента, а экспериментальная информация используется для выбора математической модели из некоторого заданного множества таких моделей.

Результатом функционирования АСНИ является подтверждение (отклонение) гипотез или совокупность законченных математических моделей, удовлетворяющая заданным требованиям, а также обработанные результаты исследований, наблюдений и измерений.

Функционирование АСНИ должно обеспечивать получение выходных документов, выполненных в заданной форме и содержащих результаты науч-

ных исследований или испытаний, а также рекомендации по использованию этих результатов для прогнозирования, управления или проектирования.

13.2.4. Структура АСНИ

Основными структурными звеньями АСНИ являются подсистемы.

Подсистемой АСНИ называется выделенная по некоторым признакам часть АСНИ, обеспечивающая выполнение определенных автоматизированных процедур исследований (испытаний) и получение соответствующих выходных документов.

Различаются объектно-ориентированные (объектные) и обслуживающие подсистемы АСНИ.

Объектная подсистема осуществляет получение и обработку экспериментальных данных с некоторого объекта.

Объектными могут быть, например, подсистемы:

обработки экспериментальных данных, получаемых со специализированных установок (ускорителей, спектрометров, испытательных стендов);

обработки данных на морских судах, системы для сейсморазведки и т.п.;

коллективного пользования для куста однородных экспериментальных установок или стендов.

Обслуживающая подсистема осуществляет функции управления и обработки информации, не зависящие от особенностей исследуемого явления, объекта или процесса.

Обслуживающими могут быть, например, подсистемы:

управления АСНИ;

диалоговых процедур;

численного анализа;

планирования и оптимизации эксперимента;

ввода, обработки и вывода графической информации;

информационно-поисковых процедур.

Подсистема АСНИ состоит из компонентов, объединенных общей для данной подсистемы процедурой.

Компонентом называется элемент средств обеспечения, выполняющий определенную функцию в подсистеме АСНИ.

Структурное единство подсистемы АСНИ обеспечивается связями между компонентами различных средств обеспечения, образующими подсистему.

Структурное объединение подсистем АСНИ в систему обеспечивается связями между компонентами, входящими в подсистемы.

Средства обеспечения АСНИ состоят из компонентов:

методического обеспечения;

программного обеспечения;

технического обеспечения;

информационного обеспечения;

организационно-правового обеспечения.

Компонентами методического обеспечения являются документы, в которых изложены полностью или со ссылкой на первоисточники: теория, методы, способы, математические модели, алгоритмы, алгоритмические специальные языки для описания объектов, терминология, нормативы, стандарты и другие данные, обеспечивающие методологию научных исследований или испытаний в подсистемах АСНИ.

Из состава методического обеспечения могут выделяться компоненты математического и лингвистического обеспечения.

Компонентами программного обеспечения являются документы с текстами программ, программы на машинных носителях и эксплуатационные документы, обеспечивающие функционирование соответствующих подсистем АСНИ.

Программное обеспечение подразделяется на общесистемное и прикладное. Компонентами общесистемного программного обеспечения являются, например, операционные системы, стандартные управляющие программы

на базе операционных систем, трансляторы с алгоритмических языков и языков управления, эмуляторы.

Компонентами прикладного программного обеспечения являются программы и пакеты прикладных программ, предназначенные для осуществления процедур исследований или испытаний.

Компонентами технического обеспечения являются устройства вычислительной и организационной техники, средства и устройства связи с объектом, измерительные и другие устройства или их сочетания, обеспечивающие функционирование соответствующих подсистем АСНИ.

Совокупность компонентов технического обеспечения образует комплекс технических средств АСНИ.

Компонентами информационного обеспечения являются документы, содержащие описания стандартных процедур, типовые математические модели, основные законы, формулы, константы и другие данные, а также файлы и блоки данных на машинных носителях с записью указанных документов, обеспечивающие функционирование соответствующих подсистем АСНИ.

Совокупность компонентов информационного обеспечения образует информационную базу (базу данных) АСНИ.

Компонентами организационно-правового обеспечения АСНИ являются методические и руководящие материалы, положения, инструкции, приказы, штатные расписания, квалификационные требования, инструкции для пользователей и другие документы, обеспечивающие взаимодействие подразделений организации или предприятия при создании, эксплуатации и развитии АСНИ.

13.2.5. Основные принципы создания АСНИ

При создании и развитии АСНИ рекомендуется применять следующие принципы:

последовательное расширение сферы автоматизации научных исследований;

интеграция АСНИ;

типизация, унификация и стандартизация компонентов АСНИ;

тиражирование типовых подсистем и компонентов АСНИ;

применение единой методологии создания и развития АСНИ;

системный подход к проектированию;

адаптивность;

разработка критериев эффективности АСНИ;

ориентация на методики ведущих в тематике коллективов;

опережающее развитие базовых решений в головных организациях.

Последовательное расширение сферы автоматизации научных исследований предполагает:

внедрение средств автоматизации в новые области научных исследований, в первую очередь в те области, где получение новых существенных результатов невозможно без использования средств автоматизации;

расширение контингента пользователей автоматизированных систем научных исследований - от экспериментаторов до руководителей крупных научных программ;

автоматизация всех этапов научных исследований от планирования и управления экспериментами до анализа и перспективного планирования основных направлений научных исследований.

Тематическая, функциональная и территориальная *интеграция* АСНИ должна быть направлена в первую очередь на создание систем коллективного пользования:

для крупных экспериментальных, исследовательских и опытных установок и комплексных производственных испытаний различных технических объектов в исследовательских и проектных организациях, в ВУЗах, на предприятиях, полигонах и т.п.;

для отдельных крупных научно-исследовательских организаций, проводящих комплексные исследования сложных объектов;

для взаимосвязанных единой программой работ или родственных по тематике групп исследовательских и проектных организаций;

для территориально объединенных групп исследовательских и проектных организаций, некоторых республиканских академий наук, академических и ведомственных научных центров.

Интеграция АСНИ включает в себя:

создание многомашинных иерархических измерительно-вычислительных комплексов коллективного пользования, обслуживающих несколько экспериментов;

развитие информационной базы (создание централизованных и распределенных банков научных данных, обмен научными данными по каналам связи между АСНИ в согласованных форматах, унификацию структур данных и типизацию систем управления базами данных);

развитие общесистемного программного обеспечения (унификацию операционной среды, использование стандартных и создание специализированных телекоммуникационных методов доступа, создание многоабонентских систем реального времени, работающих в режимах мультидоступа).

В качестве основы для создания АСНИ должны использоваться типовые, проблемно-ориентированные или специализированные измерительно-вычислительные комплексы (ИВК), включающие в себя серийные средства измерительной техники, а также типовое программное обеспечение.

Особое внимание должно быть уделено типовой аппаратуре сопряжения ЭВМ с объектом исследования, созданию типовых программно-управляемых модульных систем для сбора информации и управления сложными объектами. Требования к этой аппаратуре формируются на основе соответствующих государственных и международных стандартов с тем, чтобы обеспечить максимальную совместимость технических и программных средств АСНИ, производимых различными организациями и в различных странах. Необходимо использовать стандарты КАМАК, обеспечивающие аппаратурную и программную совместимость подсистем и компонентов АСНИ.

Важнейшим условием *унификации и типизации* компонентов и подсистем АСНИ является широкое использование в них агрегатных средств измерительной и вычислительной техники, удовлетворяющих требованиям конструктивной, информационной, эксплуатационной, энергетической и других видов совместимости.

В разработке новых компонентов АСНИ необходимо широко применять аппаратную реализацию наиболее типовых функций обработки данных, операционных систем и других функций управления операционной средой.

Тиражирование типовых подсистем и компонентов АСНИ основано на типизации, унификации и стандартизации проектных решений при создании подсистем и компонентов АСНИ, что создает условия для массового промышленного производства этих компонентов.

Перспективно, например, создание и тиражирование:

типовых АСНИ для экспериментальных исследований в подразделениях научно-исследовательских организаций, высших учебных заведений и предприятий;

типовых передвижных АСНИ для полевой разведки, для научно-исследовательских судов и других подвижных объектов, а также полигонных исследований;

типовых проблемно-ориентированных измерительно-вычислительных комплексов для сбора и обработки информации на исследовательских установках и в лаборатории.

Единая методология создания АСНИ должна учитывать достижения в смежных областях науки и техники и использовать взаимное влияние тенденций развития техники, технологии и производства, с одной стороны, и автоматизации научных и производственных экспериментов - с другой. Это требование обеспечивается:

ориентацией развития автоматизации в научно-исследовательских организациях Академии наук СССР, а также министерств и ведомств на еди-

ную методологическую, техническую и программную основу технологии открытых систем;

типизацией, унификацией и стандартизацией проектных решений при создании АСНИ независимо от области их применения;

сближением принципов и технологии крупнейших научных и промышленных экспериментов;

использованием опыта создания и эксплуатации автоматизированных систем управления технологическими процессами (АСУ ТП);

разработкой однотипных методов и средств автоматизации крупных научных экспериментов, с одной стороны, и промышленных экспериментов, испытаний технических объектов и систем и опытного производства - с другой.

Системный подход в проектировании предполагает проведение проектирования на основе системного анализа, включающего решение комплекса технических, экономических, организационных вопросов, решение которых в совокупности обеспечит создание АСНИ оптимальным способом.

Адаптивность предполагает легкую приспособляемость АСНИ к изменению решаемых с ее помощью задач - scalability.

Разработка критериев эффективности АСНИ должна позволить дать объективную оценку экономического или иного эффекта, получаемого от внедрения АСНИ.

При создании или заимствовании компонентов АСНИ должны обеспечиваться требования к этим компонентам, вытекающие из общесистемных принципов, изложенных выше.

Развитие (совершенствование) компонентов АСНИ осуществляется путем создания новых модификаций (в том числе новых редакций, версий, типов) этих компонентов.

13.3. Системы Автоматизированного Проектирования

Рост доли автоматизированных производств в мировой промышленности, поставил также проблему централизованного автоматизированного управления этими производствами. Сложность реализации такой системы управления привела к тому, что специалисты в области АСУ занялись в первую очередь автоматизацией собственного рабочего места, а не автоматизацией производства в целом.

Потребность – задание формулируется в виде совокупности определенных документов. Эта совокупность - исходное описание системы. Результат проектирования - пакет документации, необходимой для изготовления изделия. Эта совокупность - окончательное описание.

Проектирование – это процесс, заключающийся в преобразовании исходного описания в окончательное, на основе выполнения работ конструкторского, исследовательского, расчетного характера.

13.3.1. Стадии проектирования

1. *Предпроектная стадия* (НИР).

Изучаются потребности, анализируются ресурсы, основные принципы построения и формируется техническое задание для изделий.

В обязательном порядке проводится обследование всех литературных источников на данную тему, проводится полное патентное исследование, и анализируются все подобные системы.

2. *Стадия эскизного проекта* (ОКР).

ОКР - опытно-конструкторские работы, проверяется корректность и реализуемость основных принципов.

3. *Стадия технического проекта*.

Выполняется всесторонняя проработка всех частей проекта и детализируются все технические решения.

4. *Стадия рабочего проекта*.

Формируется вся необходимая документация для изготовления изделий.

5. *Стадия испытаний*.

Приемочные испытания.

6. *Стадия опытной эксплуатации*.

5 и 6 стадии позволяют выявить недостатки, и уточнить технические решения.

7. *Стадия внедрения*.

Передается вся необходимая документация для выпуска готового изделия. Каждый этап делится на процедуры, а они подразделяются на операции.

Различают две технологии проектирования:

восходящее проектирование;

нисходящее проектирование.

Производится унификация отдельных процедур по изготовлению отдельных узлов, элементов, которые выполняются многократно. Инструментарии предназначаются для текстовой и графической информации.

Маршрут проектирования - это последовательность этапов и процедур для проектирования объекта. Маршруты для многих процессов могут быть одинаковыми. Это типовые маршруты.

Пример: построение любой БД начинается с построения информационной модели; далее - выбор СУБД; производится формирование логической структуры БД, проектирование физической структуры БД; и т.д.

13.3.2. Классификация типовых проектных процедур

Процедуры:

Синтеза

1. Структурный синтез:

-выбор структуры принципов

-выбор технических режимов

-формирование документации

2. Параметрический синтез:

•формирование технических требований устройства и назначение

•формирование параметров элементов

•идентификация математической модели.

Анализа

1. Одновариантный анализ:

- анализ статики
- анализ динамики
- анализ частотной области
- анализ устойчивости

2. Многовариантный анализ:

- чувствительность
- статистический анализ
- расчет зависимостей
- расчет выходных параметров

Нисходящую технологию проектирования начинают с унифицирования маршрутов проектирования, т.е. выбирается унифицированный маршрут, он разбивается на проектные процедуры, и эти процедуры унифицируются. Подход позволяет унифицировать решения в рамках отдельной задачи или предметной области.

Восходящая технология проектирования. Не документируется, для каких задач будут использоваться процедуры, т.е. просто создается инструментарий для автоматизации отдельных проектных процедур:

- создание и редактирование графической части
- автоматизация выполнения художественных и дизайнерских работ
- выполнения прочностных расчетов

- автоматизация проектирования текстовой части конструкторской документации.

В технологии проектирования появилась *объектно-ориентированная технология*. Это симбиоз проблемно-ориентированного и инструментального проектирования.

13.3.3. Техническое обеспечение САПР

Требования к техническому обеспечению:

•*Очень высокая производительность вычислительной техники (ВТ).* При проектировании используется оптимизационный алгоритм в том или ином виде, а это очень замедляет работу на отдельных этапах. Единственный выход - использовать высокопроизводительную вычислительную технику.

•*Требуется высокая точность расчетов.* Если требуется работа с 32-разрядными числами на машинах с меньшей разрядностью, то скорость и быстродействие очень резко падают. Выход - использовать 32-разрядные, 64-разрядные (и выше) машины с большой тактовой частотой.

•Техника должна иметь чрезвычайно *развитую периферийную аппаратуру* ("мышь", диджитайзер, сканеры и т.д.).

•Комплекс технических средств *должен позволять параллельную разработку подсистем* проектируемой системы одновременно разными конструкторами.

•Конструкторская база данных единая, а проектируют все одновременно на своих компьютерах, база данных установлена на сервере.

•Периферийная техника должна быть установлена непосредственно в подразделениях конструкторского бюро. Проще всего защитить информацию и технику, если *все собрать в ВЦ*.

•ВТ должна обеспечивать *необходимый уровень секретности и защиты информации.*

•Носители должны соответствовать характеру используемой информации.

13.3.4. Структура комплекса технических средств и его состав

1. *Одноуровневая система*.

Наиболее распространен вариант - одна суперЭВМ с большим количеством терминалов.

2. *Многоуровневая система*.

Существует иерархия. Чаще всего на нижнем уровне - высокопроизводительные интеллектуальные терминалы (ПК). На втором уровне ставятся миниЭВМ, на верхнем уровне - большие, суперЭВМ.

3. *Двухуровневая система*.

На верхнем (втором) - суперЭВМ, на нижнем- индивидуальные терминалы, связь между которыми осуществляется на уровне файлов.

Устройства для ввода пространственной информации

Бывают двух типов:

- механические
- оптические.

Механические основаны на принципе отображенного луча, на основе лазерной техники.

Оптические средства - постоянное вращение объекта и с шагом перемещают источник света, снимая отображенные лучи. Эти устройства чрезвычайно неточные и дорогие. Отечественных нет.

Устройства вывода

Особенности:

- Все устройства вывода имеют область рисования и область вывода. Область рисования определяется физическими и техническими характеристиками устройства. Для управления устройством вывода пишется драйвер. Драйвер пишется для конкретного пакета прикладных программ (ППП). Драйвер ориентируется на общепринятый стандарт, и в нем программным путем задается область вывода, которая либо равна, либо меньше области рисования. Драйвер ориентируется на стандарты в стране, в мире (А4, А3, А1, А0). Если

форматы не совпадают, то это затрудняет работу. Связаны с расходным материалом.

- Для эскизных чертежей пригодны устройства, позволяющие использовать стержень шариковой ручки (прокатывать два раза по месту). Толстая линия за 4, 5 проводов. Скорость и качество снижается. Более качественно рисует фломастер. Он позволяет рисовать с максимальной скоростью. Недостатком является то, что у фломастера короткий срок службы. Фломастер не выдерживает толщину линий. Исключение - керамические фломастеры.
- Наиболее точно и качественно позволяют работать рапитографы. Рисуют медленнее всех.
- Универсальное средство - карандашные графопостроители. Непрерывная линия, можно менять яркость линий, цвета.
- Тип бумаги для различных принтеров свой. В паспорте указан тип бумаги, на который он дает качественное изображение. У рулонных графопостроителей нужно 10 см, чтобы захватить бумагу, т.е. формат удлиняется на 10 см. Бумага нужна жесткая.
- САПР требует в качестве средств вывода устройства быстрой печати (400-1200 строк в минуту)

Информационное обеспечение САПР

Особенности:

1. Чрезвычайно большой объем нормативны баз. Следствием этого является наличие носителей большого объема (винчестеры 200, 600 Мбайт) и высокоскоростные винчестеры.
2. Очень большой объем графической информации. Должен быть быстрый доступ к графическим объектам. Все чертежи должны храниться в разархивированном виде.
3. При конструировании для развития графической базы все вновь разрабатываемые графические объекты подлежат хранению. База - сот-

ни Гбайт. Используют лазерные носители. Выходы - за счет применения носителей с большим объемом запоминающих устройств (ЗУ) - оптические диски, стриммеры. Использование в качестве устройства поиска аппарата для поиска микрофиш.

4. САПР практически не использует одного типа БД. Это древовидные БД нормативно-справочной информации, реляционные базы для условно-постоянной и оперативной информации, базы в виде последовательных файлов для хранения графической информации; реляционные базы для правил вывода, построения, для хранения баз знаний.

Программное обеспечение САПР

Особенности:

1. Программное обеспечение должно позволять сетевую работу в рамках САПР. Программное обеспечение чаще всего имеет не один пакет ПП, а состоит из очень большого количества ППП.

В рамках САПР ни одна проблемно-ориентированная подсистема - совокупность программного, информационного, технического, организационного и других видов обеспечений. Подсистему обычно размещают на одном компьютере.

2. Программное обеспечение САПР чрезвычайно дорогое. Чаще САПР разрабатывают для группы предприятий. САПР должна легко подстраиваться под конкретного пользователя. Выход: так как период адаптации системы иногда превышает время разработки новых систем, выбирают путь разработки инструментальных систем. Инструментальные системы строятся так: формируется модель предметной области. На основе задания на проектирование начинают строить решение задачи. Для этого задание, используя п.1 и 2, проходит трансляцию на внутренний язык. Оттранслированное задание передается системе "Монитор". В рамках монитора модуль анализирует задание на внутреннем (промежуточном) языке и строит алгоритм его реализации.

Далее планировщик подбирает модули из ПОП и ППП для реализации любой операции проектирования, после чего управление передает компоновщику.

Компоновщик, зная языки ПОП, устанавливает интерфейс между модулями и собирает модули в отдельную программу. После чего проводится анализ технических средств пользователя и генерируется программа, позволяющая решать поставленную задачу.

ОО САПР

Отличается тем, что, кроме заказчика, в роли которого обычно министерство, в объединении имеется еще пользователь, в функции которого входит:

1. участие в предпроектном обследовании
2. участие в разработке технического задания
3. адаптация готового ПО для своих условий
4. формирование штата сотрудников системы
5. подготовка производства к внедрению САПР
6. участие в проведении испытаний и приемке системы

подготовка и переподготовка кадров

Тема 14. Основы защиты информации – 2 часа

14.1. От чего защищают информацию

Чаще всего информацию *защищают* от:

1.физической утраты;

2.нежелательного изменения, удаления, порчи пользователем;

3.удаления, изменения, порчи в результате использования бракованного программного обеспечения;

4.порчи, изменения, удаления компьютерными вирусами и подобными им программами;

5.несанкционированного изменения, просмотра, удаления посторонними лицами.

Физическая утрата информации может произойти из-за

- выхода из строя (износа, поломки) носителя информации,
- кражи компьютера и/или носителя информации,
- стихийного бедствия (пожара, наводнения и т.д.),
- нарушения правил эксплуатации вычислительной техники и носителей информации.

Меры по предотвращению физической утраты информации:

- резервное копирование ценной информации;
- хранение и установка вычислительной техники (и носителей информации) в охраняемых и защищенных от внешних воздействий помещениях (возможно использование несгораемых сейфов и т.п.);
- строгое соблюдение правил хранения и эксплуатации вычислительной техники и носителей информации.

Нежелательное удаление, изменение, порча информации пользователем может произойти из-за невнимательности при работе с компьютером, случайного нажатия клавиш и т.д. Для предотвращения вышеупомянутых действий следует быть внимательным при работе с компьютером, по возможности исключить случайное нажатие клавиш, создавать резервные копии важных данных, настроить используемое программное обеспечение таким образом, чтобы перед удалением, внесением изменений в информацию обязательно выводился запрос на подтверждение.

Во избежание *утраты информации* в результате использования *бракованного программного обеспечения* и программ, содержащих ошибки и недоработки, следует создавать резервные копии ценной информации, использовать лицензионное программное обеспечение, следить за появлением "заплаток" для программ и при необходимости устанавливать их, использовать "проверенные временем" программы.

Для предотвращения доступа к информации *посторонних лиц* следует использовать различные пароли и коды доступа, установить программное обеспечение, предоставляющее широкие возможности по защите информации.

14.2. Антивирусная защита

Методы защиты от компьютерных вирусов

Существуют *три рубежа защиты* от компьютерных вирусов:

- предотвращение поступления вирусов;
- предотвращение вирусной атаки, если вирус все-таки поступил на компьютер;
- предотвращение разрушительных последствий, если атака все-таки произошла.

Существует *три метода реализации защиты*:

- программные методы защиты;
- аппаратные методы защиты;
- организационные методы защиты.

В вопросе защиты ценных данных часто используют бытовой подход: "болезнь лучше предотвратить, чем лечить". К сожалению, именно он и вызывает наиболее разрушительные последствия. Создав бастионы на пути проникновения вирусов в компьютер, нельзя положиться на их прочность и остаться неготовым к действиям после разрушительной атаки. К тому же, вирусная атака - далеко не единственная и даже не самая распространенная причина утраты важных данных. Существуют программные сбои, которые могут вывести из строя операционную систему, а также аппаратные сбои, способные сделать жесткий диск неработоспособным. Всегда существует вероятность утраты компьютера вместе с ценными данными в результате кражи, пожара или иного стихийного бедствия.

Поэтому создавать систему безопасности следует в первую очередь "с конца" - с предотвращения разрушительных последствий любого воздействия, будь то вирусная атака, кража в помещении или физический выход жесткого диска из строя. Надежная и безопасная работа с данными достигается только тогда, когда любое неожиданное событие, в том числе и полное

физическое уничтожение компьютера не приведет к катастрофическим последствиям.

14.2.1. Компьютерные вирусы

Компьютерный вирус – это программный код, встроенный в другую программу, или в документ, или в определенные области носителя данных и предназначенный для выполнения несанкционированных действий на несущем компьютере.

Основными типами компьютерных вирусов являются:

- программные вирусы;
- загрузочные вирусы;
- макровирусы.

К компьютерным вирусам примыкают и так называемые *троянские кони (троянские программы, троянцы).*

Программные вирусы – это блоки программного кода, целенаправленно внедренные внутрь других прикладных программ. При запуске программы, несущей вирус, происходит запуск имплантированного в нее вирусного кода. Работа этого кода вызывает скрытые от пользователя изменения в файловой системе жестких дисков и/или в содержании других программ. Так, например, вирусный код может воспроизводить себя в теле других программ - этот процесс называется *размножением*. По прошествии определенного времени, создав достаточное количество копий, программный вирус может перейти к разрушительным действиям - нарушению работы программ и операционной системы, удалению информации, хранящейся на жестком диске. Этот процесс называется *вирусной атакой*.

Самые разрушительные вирусы могут инициировать форматирование жестких дисков. Поскольку форматирование диска - достаточно продолжительный процесс, который не должен пройти незамеченным со стороны пользователя, во многих случаях программные вирусы ограничиваются уничтожением данных только в системных секторах жесткого диска, что эквива-

лентно потере таблиц файловой структуры. В этом случае данные на жестком диске остаются нетронутыми, но воспользоваться ими без применения специальных средств нельзя, поскольку неизвестно, какие сектора диска каким файлам принадлежат. Теоретически восстановить данные в этом случае можно, но трудоемкость этих работ исключительно высока.

Считается, что никакой вирус не в состоянии вывести из строя аппаратное обеспечение компьютера. Однако бывают случаи, когда аппаратное и программное обеспечение настолько взаимосвязаны, что программные повреждения приходится устранять заменой аппаратных средств. Так, например, в большинстве современных материнских плат базовая система ввода-вывода *(BIOS)* хранится в перезаписываемых постоянных запоминающих устройствах (так называемая *флэш-память*). Возможность перезаписи информации в микросхеме флэш-памяти используют некоторые программные вирусы для уничтожения данных *BIOS*. В этом случае для восстановления работоспособности компьютера требуется либо замена микросхемы, хранящей *BIOS*, либо ее перепрограммирование на специальных устройствах, называемых *программаторами*.

Программные вирусы поступают на компьютер при запуске непроверенных программ, полученных на внешнем носителе (гибкий диск, компакт-диск и т.п.) или принятых из Интернета. Особое внимание следует обратить на слова *при запуске*. При обычном копировании зараженных файлов заражение компьютера произойти не может. В связи с этим все данные, принятые из Интернета, должны проходить обязательную проверку на безопасность, а если получены незатребованные данные из незнакомого источника, их следует уничтожать, не рассматривая. Обычный прием распространения "троянских" программ - приложение к электронному письму с "рекомендацией" извлечь и запустить якобы полезную программу.

Загрузочные вирусы. От программных вирусов загрузочные вирусы отличаются методом распространения. Они поражают не программные файлы, а определенные системные области магнитных носителей (гибких и жестких

дисков). Кроме того, на включенном компьютере они могут временно располагаться в оперативной памяти.

Обычно заражение происходит при попытке загрузки компьютера с магнитного носителя, системная область которого содержит загрузочный вирус. Так, например, при попытке загрузить компьютер с гибкого диска происходит сначала проникновение вируса в оперативную память, а затем в загрузочный сектор жестких дисков. Далее этот компьютер сам становится источником распространения загрузочного вируса.

Макровирусы. Эта особая разновидность вирусов поражает документы, выполненные в некоторых прикладных программах, имеющих средства для исполнения так называемых *макрокоманд*. В частности, к таким документам относятся документы текстового процессора Microsoft Word (они имеют расширение .DOC). Заражение происходит при открытии файла документа в окне программы, если в ней не отключена возможность исполнения макрокоманд. Как и для других типов вирусов, результат атаки может быть как относительно безобидным, так и разрушительным.

14.2.2. Программные методы защиты

Существует достаточно много *программных средств антивирусной защиты* (в России наиболее популярны три антивирусные программы: AVP, Dr.Web, NAV). Они предоставляют следующие возможности:

1. Создание образа жесткого диска на внешних носителях (например, на гибких дисках). В случае выхода из строя данных в системных областях жесткого диска сохраненный "образ диска" может позволить восстановить если не все данные, то по крайней мере их большую часть. Это же средство может защитить от утраты данных при аппаратных сбоях и при неаккуратном форматировании жесткого диска.

2. Регулярное сканирование жестких дисков в поисках компьютерных вирусов. Сканирование обычно выполняется автоматически при каждом включении компьютера и при размещении внешнего диска в считывающем

устройстве. При сканировании следует иметь в виду, что антивирусная программа ищет вирус путем сравнения кода программ с кодами известных ей вирусов, хранящимися в базе данных. Если база данных устарела, а вирус является новым, сканирующая программа его не обнаружит. Для надежной работы следует регулярно обновлять антивирусную программу. Желательная периодичность обновления - один раз в две недели; допустимая - один раз в три месяца. Для примера укажем, что разрушительные последствия атаки вируса W95.CIH.1075 ("Чернобыль"), вызвавшего уничтожение информации на сотнях тысяч компьютеров 26 апреля 1999 года, были связаны не с отсутствием средств защиты от него, а с длительной задержкой (более года) в обновлении этих средств.

3. Контроль за изменением размеров и других атрибутов файлов. Поскольку некоторые компьютерные вирусы на этапе размножения изменяют параметры зараженных файлов, контролирующая программа может обнаружить их деятельность и предупредить пользователя.

4. Контроль за обращениями к жесткому диску. Поскольку наиболее опасные операции, связанные с работой компьютерных вирусов, так или иначе обращены на модификацию данных, записанных на жестком диске, антивирусные программы могут контролировать обращения к нему и предупреждать пользователя о подозрительной активности.

14.2.3. Аппаратные методы защиты

Вспомогательными средствами защиты информации являются *аппаратные средства защиты.* Так, например, простое отключение перемычки на материнской плате не позволит осуществить стирание перепрограммируемой микросхемы ПЗУ (*флэш-BIOS*), независимо от того, кто будет пытаться это сделать: компьютерный вирус, злоумышленник или неаккуратный пользователь.

14.2.4. Организационные методы защиты

Основным средством *организационной* защиты информации является *резервное копирование* наиболее ценных данных. В случае утраты информации по любой из вышеперечисленных причин жесткие диски переформатируют и подготавливают к новой эксплуатации. На "чистый" отформатированный диск устанавливают операционную систему с дистрибутивного компакт-диска, затем под ее управлением устанавливают все необходимое программное обеспечение, которое тоже берут с дистрибутивных носителей. Восстановление компьютера завершается восстановлением данных, которые берут с резервных носителей.

При резервировании данных следует также иметь в виду и то, что надо отдельно сохранять все регистрационные и парольные данные для доступа к сетевым службам Интернета. Их не следует хранить на компьютере. Обычное место хранения - служебный дневник в сейфе руководителя подразделения.

Создавая план мероприятий по резервному копированию информации, необходимо учитывать, что резервные копии должны храниться отдельно от компьютера. То есть, например, резервирование информации на отдельном жестком диске того же компьютера только создает иллюзию безопасности. Относительно новым и достаточно надежным приемом хранения ценных, но неконфиденциальных данных является их хранение в Web-папках на удаленных серверах в Интернете. Есть службы, бесплатно предоставляющие пространство (до нескольких Мбайт) для хранения данных пользователя.

Резервные копии конфиденциальных данных сохраняют на внешних носителях, которые хранят в сейфах, желательно в отдельных помещениях. При разработке организационного плана резервного копирования учитывают необходимость создания не менее двух резервных копий, сохраняемых в разных местах. Между копиями осуществляют *ротацию*. Например в течение недели ежедневно копируют данные на носители резервного комплекта А, а через неделю их заменяют комплектом Б, и т.д.

14.3. Защита информации в сетях

Существует множество причин, которые могут серьезно повлиять на работу вычислительной сети и даже привести к потере ценной информации. Среди них можно выделить следующие:

1. *Сбои оборудования*, вызванные:

- нарушениями работы сетевого кабеля;
- отключением электропитания;
- отказом дисковых систем и систем архивации данных;
- нарушением работы серверов, рабочих станций, сетевых карт.

2. *Некорректная работа ПО*, приводящая к потере или порче данных из-за:

- заражениями системы компьютерными вирусами.

3. *Несанкционированный доступ*, копирование или изменение информации - случайные или умышленные, приводящие к:

- искажению либо уничтожению данных;
- ознакомлению посторонних лиц с информацией, составляющей банковскую или финансовую тайну.

4. *Ошибки обслуживающего персонала*.

14.4. Защита информации при работе в Интернет

При работе в Интернете следует иметь в виду, что насколько ресурсы Всемирной сети открыты каждому клиенту, настолько же и ресурсы его компьютерной системы могут быть при определенных условиях открыты всем, кто обладает необходимыми средствами.

Для частного пользования этот факт не играет особой роли, но знать о нем необходимо, чтобы не допускать действий, нарушающих законодательства тех стран, на территории которых расположены серверы Интернета. К таким действиям относятся вольные или невольные попытки нарушить работоспособность компьютерных систем, попытки взлома защищенных систем, использование и распространение программ, нарушающих работоспособность компьютерных систем (в частности, компьютерных вирусов).

Работая во Всемирной сети, следует помнить о том, что абсолютно все действия фиксируются и протоколируются специальными программными средствами и информация, как о законных, так и о незаконных действиях обязательно где-то накапливается. Таким образом, к обмену информацией в Интернете следует подходить как к обычной переписке с использованием почтовых открыток. Информация свободно циркулирует в обе стороны, но в общем случае она доступна всем участникам информационного процесса. Это касается всех служб Интернета, открытых для массового использования.

Однако даже в обычной почтовой связи наряду с открытками существуют и почтовые конверты. Использование почтовых конвертов при переписке не означает, что партнерам есть, что скрывать. Их применение соответствует давно сложившейся исторической традиции и устоявшимся морально-этическим нормам общения. Потребность в аналогичных "конвертах" для защиты информации существует и в Интернете. Сегодня Интернет является не только средством общения и универсальной справочной системой - в нем циркулируют договорные и финансовые обязательства, необходимость защиты которых как от просмотра, так и от фальсификации очевидна. Начиная с 1999 года, Интернет становится мощным средством обеспечения розничного торгового оборота, а это требует защиты данных кредитных карт и других электронных платежных средств.

Принцип защиты информации в Интернете основан на том, чтобы исключить или, по крайней мере, затруднить возможность подбора *адекватного* метода для преобразования данных в информацию. Одним из приемов такой защиты является *шифрование* данных.

14.4.1. Понятие о несимметричном шифровании информации

Системам шифрования столько же лет, сколько письменному обмену информацией. Обычный подход состоит в том, что к документу применяется некий метод шифрования (назовем его *ключом*), после чего документ становится недоступен для чтения обычными средствами. Его может прочитать

только тот, кто знает ключ, - только он может применить адекватный метод чтения. Аналогично происходит шифрование и ответного сообщения. Если в процессе обмена информацией для шифрования и чтения пользуются одним и тем же ключом, то такой криптографический процесс является *симметричным*.

Основной *недостаток симметричного* процесса заключается в том, что, прежде чем начать обмен информацией, надо выполнить передачу ключа, а для этого опять-таки нужна защищенная связь, то есть проблема повторяется, хотя и на другом уровне. Если рассмотреть оплату клиентом товара или услуги с помощью кредитной карты, то получается, что торговая фирма должна создать по одному ключу для каждого своего клиента и каким-то образом передать им эти ключи. Это крайне неудобно.

Поэтому в настоящее время в Интернете используют *несимметричные* криптографические системы, основанные на использовании не одного, а двух ключей. Происходит это следующим образом. Компания для работы с клиентами создает два ключа: один – *открытый (public - публичный)* ключ, а другой – *закрытый (private - личный)* ключ. На самом деле это как бы две "половинки" одного целого ключа, связанные друг с другом.

Ключи устроены так, что сообщение, зашифрованное одной половинкой, можно расшифровать только другой половинкой (не той, которой оно было закодировано). Создав пару ключей, торговая компания широко распространяет *публичный ключ* (открытую половинку) и надежно сохраняет *закрытый ключ* (свою половинку).

Как публичный, так и закрытый ключ представляют собой некую кодовую последовательность. Публичный ключ компании может быть опубликован на ее сервере, откуда каждый желающий может его получить. Если клиент хочет сделать фирме заказ, он возьмет ее публичный ключ и с его помощью закодирует свое сообщение о заказе и данные о своей кредитной карте. После кодирования это сообщение может прочесть только владелец закрытого ключа. Никто из участников цепочки, по которой пересылается информа-

ция, не в состоянии это сделать. Даже сам отправитель не может прочитать собственное сообщение, хотя ему хорошо известно содержание. Лишь получатель сможет прочесть сообщение, поскольку только у него есть закрытый ключ, дополняющий использованный публичный ключ.

Если фирме надо будет отправить квитанцию о том, что заказ принят к исполнению, она закодирует ее своим закрытым ключом. Клиент сможет прочитать квитанцию, воспользовавшись имеющимся у него публичным ключом данной фирмы. Он может быть уверен, что квитанцию ему отправила именно эта фирма, и никто иной, поскольку никто иной доступа к закрытому ключу фирмы не имеет.

14.4.2. Принцип достаточности защиты

Защита публичным ключом (впрочем, как и большинство других видов защиты информации) не является абсолютно надежной. Дело в том, что поскольку каждый желающий может получить и использовать чей-то публичный ключ, то он может сколь угодно подробно изучить алгоритм работы механизма шифрования и пытаться установить метод расшифровки сообщения, то есть *реконструировать закрытый ключ*.

Это настолько справедливо, что алгоритмы кодирования публичным ключом даже нет смысла скрывать. Обычно к ним есть доступ, а часто они просто широко публикуются. Тонкость заключается в том, что знание алгоритма еще не означает возможности провести реконструкцию ключа в *разумно приемлемые сроки*. Так, например, правила игры в шахматы известны всем, и нетрудно создать алгоритм для перебора всех возможных шахматных партий, но он никому не нужен, поскольку даже самый быстрый современный суперкомпьютер будет работать над этой задачей дольше, чем существует жизнь на нашей планете.

Количество комбинаций, которое надо проверить при реконструкции закрытого ключа, не столь велико, как количество возможных шахматных партий, однако защиту информации принято считать достаточной, если за-

траты на ее преодоление превышают ожидаемую ценность самой информации. В этом состоит *принцип достаточности защиты*, которым руководствуются при использовании несимметричных средств шифрования данных. Он предполагает, что защита не абсолютна, и приемы ее снятия известны, но она все же достаточна для того, чтобы сделать это мероприятие нецелесообразным. При появлении иных средств, позволяющих-таки получить зашифрованную информацию в разумные сроки, изменяют принцип работы алгоритма, и проблема повторяется на более высоком уровне.

Разумеется, не всегда реконструкцию закрытого ключа производят методами простого перебора комбинаций. Для этого существуют специальные методы, основанные на исследовании особенностей взаимодействия открытого ключа с определенными структурами данных. Область науки, посвященная этим исследованиям, называется *криптоанализом*, а средняя продолжительность времени, необходимого для реконструкции закрытого ключа по его опубликованному открытому ключу, называется *криптостойкостью* алгоритма шифрования.

Для многих методов несимметричного шифрования криптостойкость, полученная в результате криптоанализа, существенно отличается от величин, заявляемых разработчиками алгоритмов на основании теоретических оценок. Поэтому во многих странах вопрос применения алгоритмов шифрования данных находится в поле законодательного регулирования. В частности, в России к использованию в государственных и организациях разрешены только те программные средства шифрования данных, которые прошли государственную сертификацию в административных органах, в частности, в Федеральном агентстве правительственной связи и информации при Президенте Российской Федерации (ФАПСИ).

14.5. Электронные средства защиты

14.5.1. Электронные ключи

Электронные ключи часто рассматриваются только как средство защиты от копирования. Однако эти устройства предоставляют производителям ПО и пользователям ПК множество интересных и полезных возможностей. Поэтому не удивительно, что появление на российском рынке нового семейства электронных ключей вызвало большой интерес и тех и других.

Электронные ключи построены на основе заказной микросхемы и имеют электрически программируемую энергонезависимую память. Для их программирования, как правило, не требуется никакого дополнительного оборудования - достаточно присоединить ключ к параллельному порту ПК и запустить специальную утилиту.

Защита ПО с помощью электронных ключей позволяет отказаться от жесткой привязки программ к некопируемой ключевой дискете или конкретному компьютеру, а также освобождает пользователей от ряда неудобств, возникающих при использовании других способов защиты.

Пользователь может свободно создавать резервные копии, переписывать защищенные программы с одного компьютера на другой и т.д., однако запускаться и работать эти программы будут только при подключении электронного ключа к параллельному порту компьютера.

"Общение" защищенной программы с ключом далеко не всегда ограничивается опросом последнего в момент запуска и анализом ответа с целью прервать ее несанкционированное выполнение. Многие фирмы уже не применяют простую защиту от копирования. Они блокируют часть жизненно важных функций программы, превращая такую "нелегальную" копию в демонстрационную версию. Эффект распространения подобной программы достаточно велик, поскольку пользователю нетрудно оценить ее возможности и принять решение о приобретении лицензионной версии.

С помощью электронных ключей можно организовать прокат (например, игровых программ) и абонементное обслуживание клиента, предоставляя ему, скажем, какую-либо информацию по модему. При этом в памяти ключа создается счетчик запусков игр, числа инсталляций или принятых по

модему килобайт и т.п. Исчерпав весь лимит, клиент для получения нового ресурса должен внести абонементную плату.

14.5.2. Понятие об электронной подписи

Мы рассмотрели, как клиент может переслать организации свои конфиденциальные данные (например, номер электронного счета). Точно так же он может общаться и с банком, отдавая ему распоряжения о перечислении своих средств на счета других лиц и организаций. Ему не надо ездить в банк и стоять в очереди - все можно сделать, не отходя от компьютера. Однако здесь возникает проблема: как банк узнает, что распоряжение поступило именно от данного лица, а не от злоумышленника, выдающего себя за него? Эта проблема решается с помощью, так называемой *электронной подписи*.

Принцип ее создания тот же, что и рассмотренный выше. Если нам надо создать себе электронную подпись, следует с помощью специальной программы (полученной от банка) создать те же два ключа: *закрытый* и *публичный*. Публичный ключ передается банку. Если теперь надо отправить поручение банку на операцию с расчетным счетом, оно кодируется *публичным* ключом банка, а своя подпись под ним кодируется собственным *закрытым* ключом. Банк поступает наоборот. Он читает поручение с помощью своего *закрытого* ключа, а подпись - с помощью *публичного* ключа поручителя. Если подпись читаема, банк может быть уверен, что поручение ему отправили именно мы, и никто другой.

14.5.3. Понятие об электронных сертификатах

Системой несимметричного шифрования обеспечивается делопроизводство в Интернете. Благодаря ей каждый из участников обмена может быть уверен, что полученное сообщение отправлено именно тем, кем оно подписано. Однако здесь возникает еще ряд проблем, например проблема регистрации даты отправки сообщения. Такая проблема возникает во всех случаях, когда через Интернет заключаются договоры между сторонами. Отправитель

документа может легко изменить текущую дату средствами настройки операционной системы. Поэтому обычно дата и время отправки электронного документа не имеют юридической силы. В тех же случаях, когда это важно, выполняют сертификацию даты/времени.

Сертификация даты. Сертификация даты выполняется при участии третьей, независимой стороны. Например, это может быть сервер организации, авторитет которой в данном вопросе признают оба партнера. В этом случае документ, зашифрованный открытым ключом партнера и снабженный своей электронной подписью, отправляется сначала на сервер сертифицирующей организации. Там он получает "приписку" с указанием точной даты и времени, зашифрованную закрытым ключом этой организации. Партнер декодирует содержание документа, электронную подпись отправителя и отметку о дате с помощью своих половинок "ключей". Вся работа автоматизирована.

Сертификация Web-узлов. Сертифицировать можно не только даты. При заказе товаров в Интернете важно убедиться в том, что сервер, принимающий заказы и платежи от имени некоей фирмы, действительно представляет эту фирму. Тот факт, что он распространяет ее открытый ключ и обладает ее закрытым ключом, строго говоря, еще ничего не доказывает, поскольку за время, прошедшее после создания ключа, он мог быть скомпрометирован. Подтвердить действительность ключа тоже может третья организация путем выдачи сертификата продавцу. В сертификате указано, когда он выдан и на какой срок. Если добросовестному продавцу станет известно, что его закрытый ключ каким-либо образом скомпрометирован, он сам уведомит сертификационный центр, старый сертификат будет аннулирован, создан новый ключ и выдан новый сертификат.

Прежде чем выполнять платежи через Интернет или отправлять данные о своей кредитной карте кому-либо, следует проверить наличие действующего сертификата у получателя путем обращения в сертификационный центр. Это называется *сертификацией Web-узлов*.

Сертификация издателей. Схожая проблема встречается и при распространении программного обеспечения через Интернет. Так, например, мы указали, что Web-броузеры, служащие для просмотра Web-страниц, должны обеспечивать механизм защиты от нежелательного воздействия активных компонентов на компьютер клиента. Можно представить, что произойдет, если кто-то от имени известной компании начнет распространять модифицированную версию ее броузера, в которой специально оставлены бреши в системе защиты. Злоумышленник может использовать их для активного взаимодействия с компьютером, на котором работает такой броузер.

Это относится не только к броузерам, но и ко всем видам программного обеспечения, получаемого через Интернет, в которое имплантированы "троянские кони", "компьютерные вирусы", "часовые бомбы" и прочие нежелательные объекты, в том числе и такие, которые невозможно обнаружить антивирусными средствами. Подтверждение того, что сервер, распространяющий программные продукты от имени известной фирмы, действительно уполномочен ею для этой деятельности, осуществляется путем *сертификации издателей*. Она организована аналогично сертификации Web-узлов.

2.4 Лабораторные занятия. Самостоятельная работа студентов

ЛАБОРАТОРНАЯ РАБОТА № 1

ПЕРВОНАЧАЛЬНЫЕ СВЕДЕНИЯ О СРЕДЕ WINDOWS

WINDOWS - это операционная оболочка, обеспечивающая большое количество возможностей и удобств для пользователей и программистов.

Графическая операционная среда WINDOWS была выпущена в свет в 1985 году. Появилось большое количество мощных и удобных программ, работающих в среде WINDOWS, например, Microsoft Word, Excel и т.д., что обусловило все растущую популярность WINDOWS у пользователя. А начиная с версии 3.0, созданной в 1990 г. и предоставившей дополнительные удобства пользователям, WINDOWS начала свое победное шествие, став фактически стандартом для IBM PC-совместимых компьютеров.

1. Достоинства WINDOWS

- Единый пользовательский интерфейс, т.е. единый подход для обращения к работе любой программы.
- Многозадачность, т.е. обеспечение возможности одновременного выполнения нескольких программ, переключение с одной задачи на другую, управление приоритетами выполняемых программ.

1. Совместимость с DOS -приложениями.
2. Более совершенные средства обмена данными между программами.
3. Поддержка масштабируемых шрифтов и другие.

2. Запуск WINDOWS

Часто при включении компьютера сразу загружается операционная система WINDOWS.

Замечание: Если при включении компьютера загружается операционная система DOS, то WINDOWS запускается из командной строки DOS командой WIN из любого каталога.

При запуске WINDOWS на экране будет виден курсор мыши. Вид курсора зависит от того, в какой точке экрана он расположен, от запущенной программы. При первичном запуске это стрелка. Если курсор принимает вид песочных часов, Вам нужно подождать окончания действия.

3. Вид экрана WINDOWS

3.1 Вид экрана при загрузке WINDOWS

Поверхность экрана монитора представляет собой некий **Рабочий стол**, на котором находятся **пиктограммы** - кнопки ускоренного доступа к файлу или программе. Пиктограммы можно разместить на экране произвольным образом. Для этого достаточно просто «зацепить» их мышью и переместить на желаемое место.

В нижней части экрана расположена **Панель задач**. Правая часть панели используется для системных индикаторов (время, русская или английская раскладка клавиатуры и другие). Центральная часть панели задач используется для отображения данных об активных задачах. Если запущена какая-либо программа, то на панели задач появляется пиктограмма данной программы и ее название. Нажав мышью на кнопку пиктограммы, можно перейти в данную задачу, окно программы выйдет на первый план и будет доступно для работы.

В незначительных пределах можно управлять положением и внешним видом панели задач. Прежде всего, панель можно разместить по любой границе экрана: для перемещения ее в новое положение нажмите мышью на панель и, не отпуская клавишу мыши, переместите к одной из границ экрана. Когда появится контур прямоугольника вдоль желаемой границы, отпустите клавишу мыши.

Можно увеличить ширину панели задач. Для этого зацепите мышью внешнюю границу панели и перетащите ее в новую позицию.

Панель задач очень удобна при операциях переключения между активными программами. Если загружено несколько задач, то перейти в желаемую легче всего, нажав на ее клавишу, на панели задач.

В левой части панели задач расположена кнопка «Пуск».

3.2 Вид экрана после начала работы WINDOWS

После нажатия кнопки «Пуск» на экране раскрывается **Главное меню** оболочки WINDOWS, оно содержит следующие пункты:

ПРОГРАММЫ - список установленных программ;

ДОКУМЕНТЫ - перечень 15 последних документов, с которыми проводилась работа;

НАСТРОЙКИ - программа настройки параметров работы компьютера;

ПОИСК - программа поиска файлов и документов;

СПРАВКА - загружает справочную систему;

ВЫПОЛНИТЬ - строка**,** позволяющая запустить команды WINDOWS;

ЗАВЕРШИТЬ РАБОТУ - команда выключения компьютера.

При выборе какой-либо из строк, заканчивающихся знаком треугольника, открывается дополнительное окно, содержащее меню второго уровня, и т.д.

Закрыть любое из пунктов меню можно, переместив курсор мыши за пределы меню и щелкнув по левой клавише мыши.

4. Завершение работы WINDOWS

Для завершения работы WINDOWS необходимо выбрать команду «Завершить работу» из меню «Пуск» и в появившемся окне выбрать один из вариантов.

Если выбран вариант выхода в режим DOS , то для окончательного выхода следует командой exit вернуться в режим WINDOWS,после чего выбрать вариант «Завершить работу».

Если не все используемые программы закрыты, то перед выходом необходимо завершить работу всех открытых программ нажатием комбинации клавиш «Alt-F4». Завершение работы программ сопровождается освобождением средней части панели задач от наименований этих программ.

Задание: 1. Изучить основные сведения об операционной оболочке WINDOWS.

2. Ознакомиться с видом экрана. Оформить экран по своему усмотрению, перемещая пиктограммы и панель задач.

3. Открыть Главное меню среды WINDOWS. Найти путь к одной из стандартных программ, например, калькулятор.

4. Завершить работу среды в режиме DOS, вновь загрузить среду WINDOWS.

5. Завершить работу с выключением компьютера из сети.

Контрольные вопросы

1. Каково назначение WINDOWS. Каковы основные достоинства WINDOWS в сравнении с другими операционными оболочками?

3. Как осуществить запуск WINDOWS?

4. Поясните вид экрана WINDOWS.

5. Каково назначение и содержание главного меню системы?

6. Как правильно завершить работу в WINDOWS?

Самостоятельная работа

1. ***Зависание***. Слева на Панели задач имеется кнопка ПУСК. Это элемент управления Windows, называемой *командной кнопкой*. Наведите на нее указатель мыши и задержите на некоторое время – появится *всплывающая подсказка*: Начните работу с нажатия этой кнопки.

Справа на Панели задач расположена панель индикации. На этой панели, в частности, расположен индикатор системных часов. Наведите на него

указатель и задержите на некоторое время – появиться всплывающая подсказка с показаниями системного календаря.

2. Щелчок. Наведите указатель мыши на кнопку ПУСК и щелкните левой кнопкой – над ней откроется Главное меню Windows. Меню – это один из элементов управления, представляющий собой список возможных команд. Команды, представленные в меню, выполняется щелчком на соответствующем пункте. Все команды, связанные с элементами управления, выполняется одним обычным щелчком.

Однако у щелчка есть и другое назначение. Его применяют также для выделения объектов. Разыщите на Рабочем столе значок Мой компьютер и щелкните на нем. Значок и подпись под ним изменит цвет. Это произошло выделение объекта. Объекты выделяют, чтобы подготовить их к дальнейшим операциям.

Щелкните на другом объекте, например на значке Корзина. Выделение значка Мой компьютер снимется, а вместо него выделится значок Корзина. Если нужно снять выделение со всех объектов, для этого достаточно щелкнуть на свободном от объектов месте Рабочего стола.

3. **Двойной щелчок**. Двойной щелчок применяют для использования объектов.

Например, двойной щелчок на значке, связанном с приложением, приводит к запуску этого приложения, а двойной щелчок на значке документа приводит к открытию данного документа в том же приложении, в котором он был создан. При этом происходит и запуск этого приложения. Относительно документа оно считается родительским.

В системе Windows98 с одним и тем же объектом можно выполнить много разных действий. Например, файл с музыкальной записью можно воспроизвести (причем в разных приложениях), его можно отредактировать, можно скопировать на другой носитель или удалить. Сколько бы действий ни было возможно с объектом, всегда существует одно основное действие. Оно и выполняется двойным щелчком.

Выполните двойной щелчок на значке Мой компьютер, и на экране откроется одномерное окно Мой компьютер, в котором можно увидеть значки дисков и других устройств, подключенных к компьютеру, например принтеров.

Если нужно закрыть окно, надо щелкнуть один раз на закрывающей кнопке, которая находится в правом верхнем углу окна. Закрывающая кнопка – это элемент управления, и для работы с ним достаточно одного щелчка.

4. Щелчок правой кнопкой. Щелкните правой кнопки на значке Мой компьютер, и рядом с ним откроется элемент управления, который называется контекстным меню. У каждого объекта Windows свое контекстное меню. Состав его пунктов зависит от свойств объекта, на котором произошел щелчок. Для примера сравните содержание контекстного меню объектов Мой компьютер и Корзина, обращая внимание на их различия.

Доступ к контекстному меню – основное назначение щелчка правой кнопкой. В работе с объектами Windows (особенно с незнакомыми) щелчок правой кнопкой используется очень часто.

Контекстное меню чрезвычайно важно для работы с объектами операционной системы. Выше мы говорили, что двойной щелчок позволяет выполнять только то действие над объектом, которое считается основным. В противоположность этому в контекстном меню приведены все действия, которые можно выполнить над данным объектом. Более того, во всех контекстных меню любых объектов имеется пункт Свойства. Он позволяет просматривать и изменять свойства объектов, то есть выполнять настройки программ, устройств и самой операционной системы.

5. Перетаскивание. Перетаскивание – очень мощный прием для работы с объектами операционной системы. Наведите указатель мыши на значок Мой компьютер. Нажмите левую кнопку и, не отпуская ее, переместите указатель – значок Мой компьютер переместится по поверхности Рабочего стола вместе с ним. Откройте окно Мой компьютер. Окно можно перетаскивать с

одного места на другое, если “подцепить” его указателем за строку заголовка. Так прием перетаскивания используют для оформления рабочей среды.

6. Протягивание. Откройте окно Мой компьютер. Наведите указатель мыши на одну из рамок окна и дождитесь, когда он изменит форму, превратившись в двунаправленную стрелку. После этого нажмите левую кнопку и переместите мышь. Окно изменит размер. Если навести указатель мыши на правый нижний угол окна и выполнить протягивание, то произойдет изменение размера сразу по двум координатам (по вертикали и горизонтали).

Изменение формы объектов Windows- полезное, но не единственное использование протягивания. Нередко этот прием используют для группового выделения объектов. Наведение указатель мыши на поверхность Рабочего стола, нажмите кнопку мыши и протяните мышь вправо-вниз – за указателем потянется прямоугольный контур выделения. Все объекты, которые окажутся внутри этого контура, будут выделены одновременно.

7. Специальное перетаскивание. Наведите указатель мыши на значок Мой компьютер, и нажмите правую кнопку мыши и, не отпуская ее, переместите мышь. Этот прием отличается от обычного перетаскивания только используемой кнопкой, но дает иной результат. При отпускании кнопки не происходит перемещение объекта, а вместо этого открывается так называемое меню специального перетаскивания. Содержимое этого меню зависит от перемещаемого объекта. Для большинства объектов в нем четыре пункта (копировать, переместить, создать ярлык, отменить) для таких уникальных объектов, как Мой компьютер или Корзина, в этом меню только два пункта: создать ярлык и отменить.

ЛАБОРАТОРНАЯ РАБОТА № 2

РАБОТА С ОКНАМИ WINDOWS

1.Общий вид окна

В правом верхнем углу экрана каждой прикладной программы WINDOW находятся кнопки увеличения/уменьшения и закрытия окна и программы.

Когда Вы выбираете кнопку минимизации, прикладная программа останется в памяти системы, а окно сворачивается, для возвращения в программу необходимо выбрать ее клавишу в панели задач.

В верхней части окна располагается строка заголовка, в которой выводится название прикладной программы и имя открытого файла (если таковой имеется).

В левой части строки заголовка выводится рисунок, соответствующий данной задаче. Этим же рисунком будут маркироваться и документы, созданные в программе. Под строкой заголовка обычно располагается строка меню, индивидуальная для каждой программы. Под меню часто выводится панель инструментов. Нажимая на клавишу, соответствующую той или иной операции: можно быстро выполнить ее без поиска нужной команды меню.

2. Изменение размеров окна

Если подвести курсор мыши к границе окна, то он примет вид двунаправленной стрелки. Нажав на клавишу мыши и передвигая ее, можно изменить размер окна в направлении стрелки.

3. Перемещение окна по экрану

Для этого следует поместить курсор в строку заголовка, нажать клавишу мыши, и, не отпуская ее, переместить мышь вместе с окном по экрану.

4. Расположение окон на экране различными способами

Открытые окна могут быть расположены на экране каскадом или мозаикой. Способ расположения окон мозаикой предусматривает варианты размещения окон, кроме того, по вертикали или горизонтали. Для расположения открытых окон одним из указанных способов следует поместить курсор мыши на свободный участок **панели задач** и щелкнуть правой клавишей мыши. В появившемся меню выбрать соответствующую опцию.

Замечание:

- Если окно не появляется среди расположенных окон, оно, скорее всего, было минимизировано.
- Переход по всем открытым окнам осуществляется нажатием комбинации клавиш [Alt-Tab].

5.Стандартные программы WINDOWS

5.1 Калькулятор

Принцип работы с калькулятором аналогичен обычным калькуляторам, возможны два варианта работы с калькулятором: обычный и научный. Если возникают сомнения в назначении клавиш, то следует нажать (над клавишей) правую клавишу мыши и щелкнуть правой клавишей мыши по появившейся надписи «Что это?». На экране появится необходимое описание.

Задание: Загрузить программу калькулятор, изучить работу программы.

5.2 Блокнот

Блокнот - это простейший текстовый не форматирующий редактор. Интересной возможностью является автоматическое добавление в текст редактируемого документа текущего времени и даты. Для этого необходимо в первой строке документа с первой позиции вставить команду .LOG (обязатель-

но прописными буквами). После этого при каждом открытии документа в конец текста будет добавляться строка с текущей датой.

Задание:

1. Загрузить программу блокнот.

2. Создать файл, содержащий текст на русском и английском языках.

3. Сохранить файл, задав ему имя.

4. Изучить меню программы Блокнот.

5. Ввести команду для автоматического добавление в текст редактируемого документа текущего времени и даты.

4. Выйти из программы Блокнот.

5. Открыть созданный файл через **Главное меню** Windows (Пуск, Документы).

5.3 Текстовый процессор WordPad

Текстовый процессор WordPad предназначен для создания документов с необходимым оформлением. По своим возможностям уступает редактору WinWord.

Задание: 1. Загрузить текстовый процессор WordPad.

2. Создать второй файл.

3. Изучить возможности программы.

5.4 Программа работы с графикой Paint

В строке состояния программы отображается текущее положение курсора, а также краткая подсказка по выполняемым операциям. Отмена последнего действия может быть осуществлена комбинацией клавиш [Ctrl-Z].

Задание: 1. Изучить элементы окна программы: меню, изображения инструментов, палитру цветов.

2. Открыть какой-либо из созданных ранее файлов (если таковой имеется).

3. Создать новый файл.

4. Выполнить рисунок, представить его в различных масштабах, проработать все пункты меню.

5. Выйти из программы Paint.

6. Открыть окна созданных Вами файлов.

7. Задать открытым окнам размеры и местоположение.

8. Разместить окна соответствующих программ на экране различными способами.

9. Закрыть все окна.

Самостоятельная работа

1. Загрузите программу Блокнот (Пуск / Программы / Стандартные / Блокнот). Сверните окно программы в кнопку. Вновь раскройте окно. Разверните его на весь экран и восстановите в исходных размерах. Уменьшите ширину окна. Переместите окно в другое место экрана. Просмотрите справку по программе Блокнот. Рассмотрите элементы диалогового окна Файл / Макет страницы … Измените размер бумаги, ориентацию, поля. Щелкните по кнопке Отмена. Закройте диалоговое окно и программу Блокнот.

ЛАБОРАТОРНАЯ РАБОТА № 3
КЛАВИАТУРА

Клавиатура IBM PC предназначена для ввода в компьютер информации от пользователя.

Условно все клавиши клавиатуры можно разделить на 4 группы.

1) Алфавитно-цифровые клавиши

С помощью алфавитно-цифровых клавиш можно набирать текст на русском и английском языках. Расположение латинских букв на клавиатуре компьютера, как правило, такое же, как на английской пишущей машинке, а русских букв - как на русской пишущей машинке.

Переключение клавиатуры с режима ввода английских букв на режим ввода русских букв осуществляется нажатием клавиши [Ctrl] или [Alt], или одновременным нажатием обеих клавиш [Shift], бывают и другие способы переключения.

2) Управляющие клавиши

[Entre] - предназначена для окончания ввода строки или команды, после нажатия данной клавиши происходит переход курсора на новую строку.

[Esc] - используется для отмены какого-либо действия или выхода из режима программы.

[Caps Lock] - фиксирует режим набора прописных букв. Это удобно при вводе текста, состоящего из таких букв. Повторное нажатие клавиши [Caps Lock] отменяет режим ввода прописных букв.

[Shift] - используется для ввода прописных букв и символов, располагающихся на верхнем регистре клавиатуры. Например, для ввода прописной буквы необходимо нажать клавишу [Shift] и, не отпуская ее, нужную буквенную клавишу. Аналогично, для ввода символов - необходимо нажать клавишу [Shift] и, не отпуская ее, нужную клавишу символа.

[Delete] - используется для удаления символа, находящегося справа от курсора.

[Backspace] (стрелка влево **[←]** над клавишей [Entre]) - удаляет символ, находящийся слева от курсора.

[Insert] - предназначена для переключения между двумя режимами ввода символов: ввода с раздвижкой символов (режим вставки) и ввода с замещением ранее набранных символов (режим замены).

[Home]/[End] - переводит курсор на начало/конец строки.

[Page Up]/[Page Down] - позволяет листать страницы текста вверх/вниз.

←↑→ - клавиши передвижения курсора в соответствующем направле-
↓ нии.

[SpaceBar] или [Пробел] (большая клавиша внизу клавиатуры) - позволяет выполнить пробел между словами.

[Ctrl], [Alt] - специальные клавиши предназначены для изменения других клавиш. Как и клавиша [Shift], они вводятся в комбинации с другими клавишами.

[Tab] - предназначена для ускоренного перемещения по тексту документа.

3. Функциональные клавиши

Функциональные клавиши **F1-F10** (на некоторых клавиатурах **F1-F12**) располагаются в верхнем ряду. Различные программы используют их по-разному, часто в комбинации с клавишами [Ctrl], [Alt] [Shift].

4. Дополнительная цифровая клавиатура

Дополнительная цифровая клавиатура располагается справа от основной. В зависимости от включения клавиши **[NumLock]** можно использовать ее как клавиатуру калькулятора, либо как клавиши управления курсором.

5. Особые комбинации клавиш

[Ctrl-Break] - завершение работы выполняемой программы или команды.

[Ctrl-Alt-Del] (одновременное нажатие клавиш [Ctrl], [Alt] и [Del]) - перезагрузка компьютера.

[Ctrl-C] - прекращение работы любой команды или программы DOS.

[Ctrl-S] - приостановка выполнения программ.

Задание:

1. Ознакомиться с расположением различных клавиш клавиатуры компьютера.

Контрольные вопросы

1. Как можно заменить слово (символ) в строке.
2. Как можно вставить слово (символ) в строку.
3. Как можно сдвинуть строку (слово) вправо.
4. Как можно сдвинуть строку (слово) влево.
5. Как можно одну строку разделить на две строки.
6. Как можно две соседние строки соединить в одну.
7. Как можно переключиться на русский (латинский) алфавит.
8. Как можно временно переключиться на прописные буквы.
9. Как можно постоянно вводить прописные буквы.
10. Как можно удалить символ слева от курсора.
11. Как можно удалить символ справа от курсора.

Самостоятельная работа

1. Набрать текст на русском и английском языках.

ЛАБОРАТОРНАЯ РАБОТА № 4

РАБОТА С ФАЙЛАМИ И КАТАЛОГАМИ В СРЕДЕ WINDOWS

1. Создание каталога и файла на рабочем столе WINDOWS

1.1 Создание каталога (папки) на рабочем столе WINDOWS

- наведите указатель мыши на свободное место рабочего стола и щелкните правой кнопкой мыши;
- в открывшемся меню выберите пункт «Создать» и в появившемся меню щелкните на «Папка»;
- наберите имя каталога и нажмите клавишу <Enter>.

Замечания:

1) в WINDOWS каталоги называются папками;

2) имя каталога или файла может быть записано как русскими, так и латинскими буквами и содержать до 255 символов.

Созданная таким образом папка окажется расположенной в каталоге Рабочий стол, ее можно увидеть, раскрыв этот каталог с помощью программы "Проводник" (см. панель инструментов или пункт «Программы» главного меню) или "Мой компьютер" (см. рабочий стол WINDOWS). Для помещения созданной папки в нужный каталог необходимо произвести с ней одну из предложенных операций:

- создание папки сразу в заданном каталоге (см. п. 2);
- перенос в заданный каталог (см. п.3).

1.2 Создание текстового файла на рабочем столе WINDOWS

- наведите указатель мыши на свободное место рабочего стола и щелкните правой кнопкой мыши;
- в открывшемся меню выберите пункт «Создать» и в появившемся меню щелкните на «Текстовый документ";
- наберите имя файла и нажмите клавишу <Enter>.

2. Создание папки в заданном каталоге

- откройте окно "Проводник" (см. панель инструментов или пункт «Программы» главного меню) или "Мой компьютер";
- щелкните мышью на папке, в которой нужно создать файл. Папка "раскроется";
- щелкните правой кнопкой мыши на свободном месте раскрывшейся папки;
- создайте каталог аналогично пункту 1.1.

3. Перенос папки (или файла)

- поместите каталог (файл), который нужно перенести на видное место рабочего стола с помощью мыши;
- откройте с помощью программ «Мой компьютер» или «Проводник» папку, куда нужно осуществить перенос;
- "возьмите" мышью каталог (файл) и, не отпуская левой кнопки мыши, тяните его к нужной папке, куда производите перенос. Вместе с указателем перемещается и пиктограмма каталога (файла);
- когда "принимающая" папка выделится, отпустите кнопку мыши. В результате на старом месте каталог (файл) исчезнет;
- просмотрите "принимающую" папку, раскрыв ее двойным щелчком мыши или клавишей <Enter>.

Замечание:

1. для переноса на другой диск (например, на дискету) перетаскивание осуществлять при нажатой клавише <Shift>;
2. аналогично можно перенести папку (или файл), указав на нее мышью, щелкнув правой кнопкой и выбрав пункт «Вырезать», папка (или файл) будет помещена в буфер обмена. Затем необходимо открыть ту папку, в которую необходимо перенести данную папку (или файл) и, щелкнув правой клавишей мыши на свободном месте, выбрать пункт меню «Вставить», папка (или файл) переместится в нужный каталог.

4. Копирование папки (или файла)

Выполняется аналогично переносу при нажатой клавише <Ctrl>.

Замечание:

1) для копирования на другой диск (например, на дискету) клавишу <Ctrl> нажимать не нужно;
2) аналогично можно скопировать папку (или файл), указав на нее мышью, щелкнув правой кнопкой и выбрав пункт «Копировать», папка

(или файл) будет помещена в буфер обмена. Затем необходимо открыть ту папку, в которую необходимо скопировать данную папку (или файл) и, щелкнув правой клавишей мыши на свободном месте, выбрать пункт меню «Вставить», папка (или файл) скопируется в нужный каталог.

5. Переименование папки (или файла)

•щелкните на имени файла (каталога) левой кнопкой мыши. Имя выделится. Подождите секунду и щелкните на имени еще раз. Имя обведется в рамку;

• введите новое имя, зафиксируйте его клавишей <Enter> или мышью на свободном месте рабочего стола.

Замечание: аналогично можно переименовать файл (папку), указав на нее мышью, щелкнув правой кнопкой и выбрав пункт «Переименовать».

6. Получение информации о файлах каталога

• откройте с помощью программ «Мой компьютер» или «Проводник» папку, о которой нужно получить информацию;

• через меню «Вид» или кнопки панели инструментов выберите удобный вариант представления содержимого каталога: крупными значками; мелкими; в виде списка; таблицей (в виде полной информации о записях каталога);

• прочитайте информацию об интересующем Вас файле.

Задание: 1. На рабочем столе создать папку ГРУППА ___ (где вместо символов подчеркивания ___ введите номер Вашей группы и подгуппы).

2. На рабочем столе создать файлы ЗИМА-__, ВЕСНА-__, ОСЕНЬ-__ и ЛЕТО-__.

3. В каталоге ГРУППА ___ создать папку ПРИМЕР.

4. Перенести файлы ЗИМА-___ и ВЕСНА-___ в каталог ГРУППА ___.

5. Скопировать файл ЛЕТО-___ в каталог ПРИМЕР.

6. Перенести каталог ГРУППА ___ в папку МОИ ДОКУМЕНТЫ.

7. Перенести файл ВЕСНА-___ из каталога ГРУППА ___ в каталог ПРИМЕР.

8. Скопировать файл ОСЕНЬ-___ в каталог ПРИМЕР.

9. На рабочем столе переименовать файла ОСЕНЬ-___ и ЛЕТО-___ в файлы КАНИКУЛЫ-___ и СЕССИЯ-___, соответственно.

10. Перенести файлы ОСЕНЬ-___ и ЛЕТО-___ из каталога ПРИМЕР в каталог ГРУППА ___.

11. Переименовать папку ПРИМЕР в БАКАЛАВР.

12. Просмотреть содержимое папок МОИ ДОКУМЕНТЫ, ГРУППА ___ и БАКАЛАВР в форме крупных значков, мелких значков, списка и таблицы.

13. Записать объем памяти всех созданных файлов.

14. Просмотреть с помощью программы "Проводник" и записать в тетрадь структуру (дерево) каталога МОИ ДОКУМЕНТЫ.

Самостоятельная работа

1)Открыть папку \ Мои документы (Пуск избранное мои документы).

2)Щелкните на раскрывающей кнопке разверните окно на полный экран.

3)В строке меню дайте команду файл создать папку. Убедитесь в том, что в рабочей области окна появился значок папки с присоединенной надписью новая папка.

4)Щелкните правой кнопкой мыши на свободной от значков рабочей области окна текущей папки. В открывшемся контекстном меню выберите команду создать папку. Убедитесь в том, что в пределах окна появился значок папки с надписью новая папка (2).

5)Щелкните правой кнопкой мыши на значке новая папка. В открывшемся контекстном меню выберите пункт переименовать. Дайте папке содержательное имя, например экспериментальная. Аналогично переименуйте

папку новая папка (2). Убедитесь в том, что операционная система не допускает существования в одной папке (\мои документы) двух объектов с одинаковыми именами. Дайте второй папке имя мои эксперименты.

6)Восстановите окно папки \ Мои документы до нормального размера щелчком на восстанавливающей кнопке.

7)Откройте окно Мой компьютер. В нем откройте окно с содержимым жесткого диска (C:). Пользуясь полосами прокрутки разыщите в нем папку \ Windows и откройте ее двойным щелчком. Ознакомьтесь с текстом предыдущего сообщения о том, что изменение содержания этой системной папки может быть потенциально опасным. Включите отображения содержимого папки щелчком на гиперссылке Показать файлы. В открывшемся содержимом разыщите значок папки \ Temp и откройте ее (эта папка считается папкой временного хранения данных, и экспериментировать с ней можно без опасений). Перетаскиванием переместите папку \Экспериментальная из папки Мои документы в папку C:\Windows\Temp. Специальным перетаскиванием переместите папку \Мои эксперименты в папку C:\Windows\Temp и по окончании перетаскивания выберите пункт Переместить в открывшемся контекстном меню.

8)Откройте окно C:\Windows\Temp. Щелчком выделите значок папки \Экспериментальная. При нажатой клавише CTRL щелчком выделите значок папки \Мои эксперименты. Убедитесь в том, что в рабочей области одновременно выделено два объекта (групповое выделение).

9)Заберите выделенные объекты в буфер обмена комбинацией клавиш CTRL+X. Убедитесь в том, что значки исчезли в рабочей области папки.

10)Откройте окно папки \Мои документы. Вставьте в него объекты, находящиеся в буфере обмена (CTRL+V).

ЛАБОРАТОРНАЯ РАБОТА № 5

УДАЛЕНИЕ, ВОСТАНОВЛЕНИЕ, ПОИСК И СОРТИРОВКА ЗАПИСЕЙ В WINDOWS

1. Удаление файла (каталога)

• с помощью программ "Мой компьютер" или "Проводник" найдите нужный файл (каталог);

1.1 Через меню

• укажите мышью на имя файла (каталога), щелкните правой клавишей;

• выберите пункт меню «Удалить»;

• убедитесь, что удаляете нужный файл (каталог) и подтвердите удаление.

1.2 Вручную

• укажите мышью на пиктограмму файла (каталога), нажмите на левую клавишу мыши;

• не отпуская клавишу, перетащите файл в «Корзину»;

• отпустите клавишу мыши.

2. Восстановление удаленного файла

• откройте содержимое «Корзины» двойным щелчком левой кнопки мыши или правой через меню «Открыть»;

• найдите нужный файл и, выделив мышью его имя, перетащите его на рабочий стол. То же можно выполнить через меню «Файл»/«Восстановить»;

• если необходимо, перенесите восстановленный файл в нужную папку.

3. Удаление файла из «Корзины»

Выполняется для файлов, восстановление которых не понадобится.

• откройте «Корзину»;

• укажите на нужный файл;

• выберите меню «Файл»/«Удалить» и подтвердите необходимость удаления.

4. Расстановка записей каталога в заданном порядке

• откройте нужный каталог;

• выберите меню «Вид»/«Упорядочить значки»/:

«по имени» - для расстановки по алфавиту имен;

«по типу» - для расстановки файлов различных типов группами;

«по размеру» - в порядке увеличения объема занимаемой памяти;

«по дате» - устанавливает первыми самые последние из созданных файлов.

5. Поиск файла или папки

• в главном меню выберите пункт «Поиск»/«Файлы и папки»;

5.1 Поиск по имени

• осуществляется через пункт «Имя и размещение»;

• в строке «Имя» запишите любую часть имени файла (не обязательно с начала имени) для поиска всех файлов и папок с указанным набором символов;

• нажмите кнопку «Найти».

5.2 Поиск по фрагменту информации

• осуществляется через пункт «Дополнительно»;

• в строке «Искать текст» наберите известный Вам фрагмент текста, по которому нужно найти файл;

• нажмите кнопку «Найти».

5.3 Поиск по дате создания или последнего изменения

• осуществляется через пункт «Дата изменения»;

- поставьте флажок в строке «Найти все файлы созданные либо измененные»;
- выберите один из вариантов ограничения поиска;
- нажмите кнопку «Найти».

5.4 Поиск по заданному размеру занимаемой памяти

- осуществляется через пункт «Дополнительно»;
- задайте границы размера в Кбайтах в строке «Размер»;
- нажмите кнопку «Найти».

Замечания:

1)возможно осуществление поиска при задании всех трех условий;

2)если не известно в каком каталоге находится искомый файл при поиске следует поставить метку в строке «Просмотреть все вложенные папки»;

3)при поиске можно задать форму представления найденных файлов через меню «Вид» в окне поиска.

Задание: 1. Откройте папку BP\BIN на диске C:.

2. Задайте представление содержимого папки в виде списка.

3. Расставьте записи папки по алфавиту. Убедитесь в этом.

4. Расставьте записи папки по типу. Определите отличие от предыдущего способа сортировки записей.

5. Задайте представление содержимого папки в табличном виде.

6. Расставьте записи папки по размеру занимаемой памяти. Убедитесь в этом. Запишите объем самого большого файла.

7. Расставьте записи папки по дате создания или последнего изменения. Убедитесь в этом. Определите сколько файлов создано за последние 2 дня.

8. Найдите все файлы и каталоги, имеющие в своем имени символы "___". Записать их.

9. Определите средствами поиска, сколько файлов создано за неделю после Вашего предыдущего занятия.

10. Найдите и запишите один из файлов, содержащий в своем тексте строку "Справка". Для преждевременного прекращения поиска используется кнопка «Стоп».

11. Определите сколько файлов и из каких каталогов занимают более 5000 Кбайт памяти.

12. Удалите файл ЗИМА-___ из каталога ГРУППА ___ вручную

13. Удалите файл ЛЕТО-___ из каталога ГРУППА ___ с использованием меню.

14. Удалите папку БАКАЛАВР. Откройте корзину, убедитесь в наличии в ней удаленных файлов. Обратите внимание, что в корзине нет папок, а есть только файлы.

15. Восстановите файлы ЗИМА-___, ВЕСНА-___ на рабочем столе.

16. Перенесите восстановленные файлы в папку ГРУППА ___.

17. Удалите из корзины оставшийся там файл ЛЕТО-___.

Самостоятельная работа

1)Выделите значки папок \Экспериментальная и \Мои эксперименты в папке \Мои документы. Щелкните правой кнопкой мыши и открывшемся контекстном меню выберите пункт Удалить. В открывшемся диалоговом окне подтвердите необходимость удаления объектов. Закройте окно папки \Мои документы.

2)Двойным щелчком на значке откройте окно Корзина. Убедитесь, что в нем находятся значки удаленных папок \Экспериментальная и \Мои эксперименты. Выделите оба значка. Щелкните правой кнопкой мыши в открывшемся контекстном меню выберите пункт Восстановить. Закройте Корзину.

3)Откройте окно папки \Мои документы. Убедитесь в том, что в нем восстановились значки папок \Экспериментальная и \ Мои эксперименты. Выделите оба значка. Удалите их с помощью клавиши DELETE при нажатой

клавише SHIFT. В открывшемся диалоговом окне подтвердите необходимость удаления объектов. Закройте окно папки \Мои документы.

4)Откройте окно Корзины. Убедитесь в том, что объекты, удаленные при нажатой клавише SHIFT, не поступили в Корзину. Закройте Корзину.

ЛАБОРАТОРНАЯ РАБОТА № 6

НАСТРОЙКА СВОЙСТВ МЫШИ И ОФОРМЛЕНИЯ РАБОЧЕГО СТОЛА

1. Настройка свойств мыши

1. Откройте диалоговое окно **Свойства: Мышь (Пуск ▸ Настройка ▸ Панель управления ▸ Мышь).**

2. Щелкните дважды на элементе управления **Область проверки**. Убедитесь, что при двойном щелчке элемент срабатывает, а при двух отдельных щелчках с продолжительным интервалом - нет.
3. Методом перетаскивания переместите движок **Скорость двойного нажатия** в крайнее правое положение. Убедитесь, что при этом интервал времени между двумя отдельными щелчками, составляющими двойной щелчок, чрезмерно занижен и выполнить двойной щелчок очень трудно.
4. Переместите движок в крайнее левое положение и убедитесь в том, что два отдельных щелчка интерпретируются как двойной щелчок.
5. Экспериментально выберите наиболее удобное для себя положение движка.
6. Откройте вкладку **Перемещение**.
7. Уменьшите чувствительность мыши, переместив движок **Скорость перемещения указателя** в крайнее левое положение. Щелкните на кнопке **Применить**.
8. Установите указатель мыши примерно в центре экрана. Не отрывая запястья от поверхности стола, подвигайте мышь в направлении

влево-вниз – вправо-вверх. Убедитесь в том, что указательмыши не достигает левого нижнего и правого верхнего углов экрана.

9. Переместите движок **Скорость перемещения указателя** в крайнее праввое положение. Щелкните на кнопке **Применить**.
10. Убедитесь в том, что указатель мыши можно провести от левого нижнего до правого верхнего углов экрана, не отрывая запястье от поверхности стола.
11. Экспериментально выберите наиболее удобное для себя положение движка. После каждого изменения его положения не забывайте задействовать командную кнопку **Применить**.
12. Установите все настройки в среднее (исходное) положение. Закройте диалоговое окно **Свойства:Мышь**.

Замечание

Операционная система Windows позволяет каждому пользователю индивидуализировать настройку органов управления. Необходимые для этого средства можно найти в окне **Панель управления (Пуск ▸ Настройка ▸ Панель управления).**

.

2. Настройка оформления Рабочего стола

1. Щелкните правой кнопкой мыши на свободном месте *Рабочего стола***.** Выберите в контексном меню пункт **Свойства** – откроется диалоговое окно **Свойства:Экран.** Данное окно можно также открыть через *Главное меню* Windows **(Пуск ▸ Настройка ▸ Панель управления ▸ Экран).**
2. Убедитесь в том, что окрыта вкладка **Фон**. В списке **Рисунок рабочего стола** выберите любой рисунок. Щелкните на кнопке **ОК**. Убедитесь в том, что фон *Рабочего стола* изменился.
3. Установите, как влияют на оформление *Рабочего стола* различные способы расположения фонового рисунка с помощью раскрывающего списка **Поместить**.

4. С помощью поисковой системы установите, где хранятся фоновые рисунки *Рабочего стола*. Для этого в поле **Имя** введите название объекта (например, Волны).
5. Откройте данную папку и найдите нужный объект. Определите, в каком формате хранятся фоновые рисунки *Рабочего стола* (тип расширения).
6. Для того, чтобы использовать для оформления Рабочего стола другие рисунки нажмите кнопку **Обзор…** в диалоговом окне **Свойства:Экран**, найдите в других папках файлы с рисунками.
7. Откройте вкладку **Заставка** в диалоговом окне **Свойства:Экран,** познакомьтесь с оформлением заставки *Рабочего стола*.
8. Установите все настройки в исходное положение.

3. Применение справочной системы Windows

1. Запустите справочную систему (**Пуск ▸ Справка**).
2. Ознакомьтесь с разделами справочной системы.
3. С помощью вкладки **Поиск** разыщите статью, в которой описан порядок действий при «выключении» компьютера.
4. Подготовьте краткий отчет о порядке действий при выключении компьютера.

Замечание

В некоторых открытых окнах в правом верхнем углу окна располагается значок **?**, с помощью которого можно получить необходимую подсказку.

5. Откройте диалоговое окно настройки экрана **(Пуск ▸ Настройка ▸ Панель управления ▸ Экран).**
6. Для получения справки установите курсор на значок **?** и перетащите его при нажатой правой клавише мыши в нужное место, откроется окно с необходимой подсказкой.

Самостоятельная работа

1. Самостоятельно настройте оформление **Рабочего стола** вашего компьютера.
2. Самостоятельно введите ключевое слово для поиска и найдите нужные разделы справочной системы.

ЛАБОРАТОРНАЯ РАБОТА № 7

ПРОВЕРКА И ДЕФРАГМЕНТАЦИЯ ДИСКА

1. Проверка диска

Для проверки логической и физической структуры диска используется программа ***Проверка диска***. Запустите ее (Пуск / Программы / Стандартные / Служебные / Проверка диска или Мой компьютер / Диск A: / Свойства / Сервис / Выполнить проверку). Установите полную проверку диска A:, просмотрите настройки (Настройка …) и установки дополнительных параметров проверки (Дополнительно …), проверьте дискету. При необходимости воспользуйтесь справкой (Пуск / Справка / Содержание / Стандартные программы / Служебные программы / Проверка диска).

2. Дефрагментация диска

Дефрагментация используется для оптимизации размещения файлов на диске, что позволяет ускорить выполнение программ. Опробуйте дефрагментацию, как и проверку, на Вашей дискете. Запустите программу дефрагментации диска A: одним из способов (по аналогии с программой проверки диска), для наглядного наблюдения за процессом в появившемся диалоговом окне щелкните по кнопке Сведения, включите Легенду – условные обозначения на схеме.

3. Антивирусные средства

Антивирусные средства, как и рассматриваемые далее программы архивации, не входят в состав операционной системы Windows, однако они необходимы для нормальной работы пользователя.

Для проверки диска на наличие программ – вирусов, которые могут вызвать самые разнообразные нежелательные эффекты в работе компьютера, используется антивирусная программа **AVP** (программа Е.Касперского). Если на компьютере установлена программа AVP-монитор, проверка происходит автоматически. Для непосредственной проверки предназначена программа AVP-сканер. Запустите ее, используя ярлык, или кнопку Пуск (при необходимости организуйте поиск программы по символам AVP). Прочитайте справку к программе. Просмотрите возможные изменения настроек, используя вкладки диалогового окна программы. Проверьте дискету, установив флажок Флоппи-дисководы и щелкнув по кнопке Пуск. При необходимости обеспечьте лечение дискеты. Просмотрите статистику отчета.

Другое широко распространенное средство антивирусной проверки дисков – программа **DrWeb**. Запустите программу и проверьте Вашу дискету с помощью этой программы.

4. Сжатие файлов

С целью экономии дискового пространства при хранении и передаче файлов используются различные методы сжатия, или ***архивация*** файлов. Наиболее распростаненной программой архивации для Windows является программа **WinZip**.

Пусть, например, требуется сохранить в сжатом виде три файла, записанных на дискету на предыдущем занятии (Мой файл.txt, Формат.doc и Мой рисунок.bmp). Определите суммарный размер этих трех файлов. Загрузите программу WinZip, щелкните по кнопке New (создать новый архив), выберите диск A: (место создания архива), убедитесь, что в поле Тип файла установлен Zip files, введите имя архивного

файла, в котором в сжатом виде будут храниться все три наши файла, например, Мой архив; проверьте наличие флажка у Add Dialog (диалоговое окно сжатия). В диалоговом окне выделите три файла, которые надо сжать (для одновременного выделения группы файлов держите нажатой клавишу **Ctrl**), и щелкните по кнопке Add. После создания архивного файла в окне WinZip появляется перечень его компонентов, их новые размеры и степень сжатия. Какой из файлов сжат в наибольшей степени? Закройте WinZip, просмотрите содержание диска A:, используя Мой компьютер. Какой размер имеет архивный файл? Каково его расширение? Обратите внимание на значок архива. Теперь удалите исходные три файла – они сохранены в архивном файле Мой архив.

Для просмотра содержимого архивного файла (если Вы не знаете, что находится в архиве) используется команда Открыть (в контекстном меню).

Для извлечения файлов из архива (распаковки) в контекстном меню архивного файла выберите Extract to … и нажмите кнопку Extract: исходные файлы появятся на дискете.

Самостоятельная работа

1. Создайте на дискете папку LAB_WIN (Лабораторные работы по операционной системе Windows) и перенесите в нее все ранее созданные файлы. Определите размер папки. Создайте архивный файл, содержащий всю папку. Определите его размер. Удалите папку LAB_WIN, распакуйте архив. Проверьте логическую и физическую структуру диска, проконтролируйте диск на отсутствие вирусов.

2. Если на компьютере установлен архиватор WinRar, попытайтесь самостоятельно разобраться с его работой: проделайте те же действия, что и с WinZip. Сравните полученные архивы по размеру (степени сжатия).

ЛАБОРАТОРНАЯ РАБОТА № 8

РЕЗЕРВНОЕ КОПИРОВАНИЕ ДАННЫХ И ПРОВЕРКА ДИСКА

1. Резервное копирование данных

6. Запустите программу Проводник (Пуск Программы Проводник). Откройте папку \Мой документы. Дайте команду Файл Создать Папку. Переименуйте созданную папку, присвоив ей имя Эксперимент. Скопируйте в папку \Эксперименты файлы dict.doc и scheme.bmp.
7. Запустите программу резервного копирования (Пуск Программы Стандартные Служебные Архивация данных).
8. Для создания задания архивации щелкните на кнопке Мастер создания архивов на панели инструментов.
9. Установите флажок Архивация выбранных файлов и щелкните на кнопке Далее.
10. На левой панели окна мастера разверните структуру папок диска С: и папку \Мои документы. Установите флажок у имени папки \Эксперимент и щелкните на кнопке Далее.
11. Установите переключатель Новые и измененные файлы и щелкните на кнопке Далее.
12. Укажите имя архива как A:\MyBackup.quc. Щелкните на кнопке Далее.
13. Установите флажки, определяющие сравнение файлов после завершения архивации и разрешающие сжатие файлов в архиве. Щелкните на кнопке Далее.
14. Вводите имя задания архивации как Эксперимент, вставьте чистый диск в дисковод А: и щелкните на кнопке Запуск. По отчету программы архивации зафиксируйте время, потраченное на архивацию.
15. Переключитесь в программу Проводник и очистите папку \Эксперимент.
16. Вернитесь в программу архивизации, выберите вкладку Восстановление и щелкните на кнопке Мастер восстановления файлов. Выберите только что созданный архив и щелкните на кнопке Далее.

17. Мастер автоматически выберет файлы, расположенные на резервной копии. Щелкните на кнопке OK.
18. Установите флажок у диска C:, чтобы показать, что требуется восстановить все файлы, имеющиеся в архиве. Щелкните на кнопке Далее.
19. Щелкните на кнопке Далее, чтобы показать, что файлы должны быть восстановлены в их исходном месте.
20. Щелкните на кнопке Запуск, а затем на кнопке OK. Просмотрите отчет о восстановлении файлов.
21. Убедитесь, что “утраченные” файлы в папке \Эксперимент восстановлены.

2. Проверка жесткого диска

1. Запустите программу проверки диска (Пуск Программы Стандартные Служебные Проверка диска).
2. Выберите диск C: в списке дисков.
3. Щелкните на кнопке Дополнительно и ознакомьтесь с настройками. Используемые программой при проверке логической структуры файловой системы. Закройте диалоговое окно Дополнительные параметры настройки диска щелчком на кнопке Отмена.
4. Установите переключатель Полная и щелкните на кнопке Настройка.
5. Ознакомьтесь с настройками, используемыми программой при проверке магнитной поверхности диска. Закройте диалоговое окно Режим проверки поверхности диска щелчком на кнопке Отмена.
6. Установите переключатель Стандартная. Сбросьте флажок Исправлять ошибки автоматически.
7. Щелкните на кнопке Запуск и наблюдайте за ходом проверки.
8. При обнаружении ошибок на диске ознакомьтесь с сообщением об ошибке и выберите среди предлагаемых вариантов исправления тот, при котором ошибка игнорируется и работа продолжается дальше.

9. Закройте отчет о результатах проверки. Закройте программу Проверка диска щелчком на кнопке Закрыть.

Самостоятельная работа

1. Самостоятельно сделайте резервную копию своих файлов.

2. Осуществите проверку жесткого диска на домашнем компьютере, подготовьте отчет о результатах проверки.

ЛАБОРАТОРНАЯ РАБОТА № 9

ПРОГРАММА FAR Manager

1. Общая характеристика оболочки.

Весьма популярным файл-менеджером является программа FAR file and archive manager. Обладающая уникальными характеристиками и возможностями, она уже долгое время занимает одно из ведущих мест среди приложений подобного рода.

Процедуры копирования, переноса, переименования унифицированы как для файлов, так и для папок: вы выбираете файл или выделяете группу файлов (папок) в одном рабочем окне, определяете, какая директория будет являться приемником, и выполняете необходимую процедуру. Многие действия можно выполнять с помощью «мыши». Интерфейс программы сильно напоминает Norton Commander.

2. Запуск программы

Сама программа Far Manager запускается из стартового меню операционной системы Windows, либо по двойному нажатию мышкой на иконке Far Manager на рабочем столе. Это приложение ориентировано на работу не только с мышью, но и с функциональными клавишами.

Для перемещения по “окнам” программы используются стандартные клавиши управления курсором и клавиши Page Down, Page Up, Home, End,

Стрелки служат для перемещения курсора в активном " окне" . Клавиши {PgUp} и {PgDn} служат для постраничному перемещению по тексту со-

ответственно вверх и вниз (перелистывание экранных страниц). Клавиши {Home} и {End} предназначаются для перемещения курсора в начало и конец строки соответственно. Клавиша {Ins} выполняет подсветку (выбор) программ для последующего копирования удаления и т.д. Для отмены подсветки достаточно повторно нажать на {Ins}. Клавиша {Del} служит для удаления символа, на котором находится курсор. Последний при этом остается на прежнем месте, а символы справа от курсора сдвигаются на одну позицию влево. Для перехода в другую папку достаточно подвести " подсветку" (highlight) к имени папки (папки изображаются в окне заглавными буквами) и нажать клавишу {Enter}. После этого в окне будет отображено содержимое данного папки. Содержимое второго окна останется без изменения - это окно НЕ активно. Для смены активного окна (т.е. окна, в котором производится работа) достаточно нажать клавишу {Tab}.Эта операция меняет активное окно. Для возврата в папку предыдущего уровня надо подвести подсветку к имени папки предыдущего уровня и нажать {Enter} или {Ctrl - PgUp} (первый способ не срабатывает, если командная строка не пуста, второй же способ игнорирует заполненность командной строки).

Серая клавиша {+} позволяет выполнить подсветку группы файлов по определенному признаку. Признак задается с использованием квазисимволов “*” и “?”.

Серая клавиша {-} позволяет отменить подсветку группы файлов активного окна.

3. Функциональные клавиши F1 - F10

Функциональные клавиши F1 - F10 расположены в нижней строке экрана и обеспечивают выполнение основных действий с файлами и папками. Рассмотрим их.

{F1} - вызов на экран помощи.

{F2} - вызов на экран меню пользовательских программ. С помощью этого меню, можно легко вызвать любую программу с диска, в каком бы подпапкае она ни находилась.

{F3} - просмотр содержимого файла в символьном виде. Эта команда выводит на экран в символьном виде содержимое файла, на который указывает подсветка.

{F4} - редактирование файла. С помощью этой команды можно осуществить простейшее редактирование файла. {F6} - перемещение файла из одного подпапкаа (дисковода) в другой. Действует аналогично команде {F5} с той лишь разницей, что в подпапкае, откуда файл переносится, он стирается.

{F7} - создать папку.

{F8} - стереть файл (файлы).

{F9} - вызов меню управления режимами Far Manager.

Работа с группами файлов

Предположем, что на диске {**C:**} имеется некая папка {**abc1**}, содержащая десять файлов, пять из которых нужно скопировать {**F5**} на диск {**A:**}, в папку {**abc2**}, а затем удалить {**F8**} их оттуда. Для этого нужно нажатием клавиши {**Ins**} или правой кнопкой “мыши”, пометить нужные файлы (они будут выделены жёлтым цветом), после чего можно перейти к операциям копирования или удаления.

Самостоятельная работа

1. Создать на диске **C:** папку **FIRST** и в ней папку **SECOND**. В папке **SECOND** открыть новый файл **frs.txt**, в который записать символы **ABCDEF**, переименовать файл в **snd.txt**, посмотреть полученный файл, удалить его, а затем и папки **FIRST** и **SECOND**.

2. Для перехода на дисковод **C:** , например, на левой панели, нажать **Alt+F1** и, выбрав **C** – **Enter**. При необходимости перейти на правую панель, нажать **Tab**. Для создания папки **FIRST** воспользоваться клавишей **F7** и ввести в открывшееся окно имя папки, нажать **Enter**. Войти в папку **FIRST**, установив курсор на ее имя, и нажать **Enter**. Вновь нажать **F7** для создания **SECOND**. Войдя в **SECOND**, открыть новый файл, для чего нажать **Shift+F4**

и ввести имя **frs.txt**, нажать **Enter**. Набрать символы **ABCDEF** и закрыть файл, сохранив его в **SECOND** (**Esc**, затем **Enter**).

3. Для переименования файла **frs.txt** на **snd.txt** на правой панели открыть папку **SECOND** (в результате на обеих панелях будет выведена одинаковая информация о его содержании). Установить курсор на имя файла **frs.txt** и нажать **F6**. В появившемся окне перейти на конец строки ввода (**End**), ввести новое имя файла **\snd.txt** и нажать **Enter**.

4. Для просмотра файла **snd.txt** установить курсор на его имя и нажать **F3**. Окончание просмотра – **Esc**, удаление – **F8**, при этом надо подтвердить намерение удалить его (снова **Enter**). Для удаления папки **SECOND** – выйти из него (курсор стоит на **..** , нажать **Enter**, курсор устанавливается на имя папки **SECOND**) и нажать **F8**. Подобным образом удалить и папку **FIRST**.

ЛАБОРАТОРНАЯ РАБОТА № 10

ПРОГРАММА БЛОКНОТ

Загрузите Блокнот (Пуск / Программы / Стандартные / Блокнот) – редактор, предназначенный для создания простых текстовых документов с сохранением их в виде текстового файла (с расширением ***txt***) и просмотра таких документов. После загрузки открывается новый документ, которому автоматически присваивается имя Безымянный. Просмотрите меню программы. В рабочей области найдите ***курсор*** редактора (вертикальная черта), указывающий положение вводимого с клавиатуры символа. Введите с клавиатуры в одну строку свои фамилию, имя и отчество. При необходимости вспомните, как переключить раскладку клавиатуры с английской на русскую. Для ввода ***заглавных букв*** нажмите клавишу **Shift**. При ошибочном написании используйте клавиши ***удаления***: нажатие **Backspace** удаляет символ слева от курсора, нажатие **Delete** – справа. Для ***перемещения*** по тексту используйте мышь или клавиши со стрелками, а также **Home** – переход в начало строки и **End** – переход в конец строки.

Разделите полученную строку на три: вначале установите курсор на первую букву имени, нажмите клавишу **Enter**, затем – на первую букву отчества, снова нажмите Enter. Для восстановления текста в виде одной строки установите курсор в конец верхней строки (End) и нажмите Delete. Снова разделите текст на три строки.

Предположим, что фамилию надо переставить в конец текста. Для этого сначала ее выделяют в виде ***фрагмента*** (блока): указатель мыши устанавливается на начало выделяемого текста и протаскивается до его конца. При необходимости это же можно проделать с помощью клавиатуры: для выделения используется сочетание клавиши Shift и стрелок в нужном направлении. Опробуйте работу обоих способов. Выделенный фрагмент вырезают в буфер, используя контекстное меню Вырезать. Затем щелчком мыши курсор устанавливают в то место, куда должен быть перенесен выделенный фрагмент, и вставляют фрагмент из буфера, используя меню Вставить. Подобным образом фрагмент может быть скопирован. Обратите внимание на полную аналогию выполнения этих операций с соответствующими операциями с файлами и папками. Перенесите фамилию в конец текста, используя операции с фрагментами. Опробуйте эти действия как в случае трех строк текста (для установки курсора в новую строку нажмите End – переход в конец последней строки, затем Enter), так и в случае, когда весь текст записан в виде одной строки.

Сохраните на дискете полученный текстовый документ. Для этого в меню Файл выберите ***Сохранить как***, в появившемся окне в поле ***Папка*** раскройте список щелчком по стрелке и выберите Диск 3.5 (A:), в поле ***Имя файла*** вместо Безымянный введите имя Мой файл и щелкните по кнопке Сохранить. При необходимости воспользуйтесь ***контекстной подсказкой***, используемой в диалоговых окнах: в строке заголовка щелкните по знаку вопроса, указатель мыши принимает форму стрелки со знаком вопроса. Щелкните по любому элементу, о котором надо получить справку: ря-

дом с ним появится контекстная подсказка. Посмотрите подсказку к элементам диалогового окна Папка, Тип файла, Сохранить и другим. Закройте Блокнот.

Используя Мой компьютер, просмотрите содержание Вашей дискеты, найдите Мой файл, обратите внимание на значок (это стандартный значок ***текстового*** файла). Для просмотра файла используйте команду контекстного меню ***Быстрый просмотр***. Проанализируйте свойства файла. Создайте к нему ярлык на Рабочем столе. Загрузите файл с помощью ярлыка. Закройте его.

Самостоятельная работа

1. Запустите текстовый редактор Блокнот (Пуск Программы Стандартные Блокнот).
2. Убедитесь, что включена русская раскладка клавиатуры. В противном случае щелкните на указателе языка на панели индикации и выберите в открывшемся меню пункт Русский.
3. Введите с клавиатуры слово Конденсатор (при вводе заглавной буквы удерживайте нажатой клавишу SHIFT) и нажмите клавишу ENTER.
4. Далее введите с клавиатуры термины Резистор, Катушка индуктивности, Выключатель, Амперметр и Вольтметр, нажимая после ввода каждого термина клавишу ENTER.
5. Расставьте в документе термины по алфавиту, выделяя строки и перемещая их через буфер обмена. Дважды щелкните на слове Амперметр и убедитесь, что оно при этом выделяется (в программе Блокнот этот способ служит для выделения отдельных слов). Нажмите комбинацию клавиш SHIFT+ВПРАВО, чтобы включить в выделенный фрагмент невидимый символ конца строки – курсор при этом переместиться в начало следующей строки.

6. Дайте команду Правка Вырезать, чтобы забрать выделенный фрагмент в буфер обмена. Убедитесь, что он действительно удаляется из документа.
7. Нажмите комбинацию клавиш CTRL+HOME, чтобы установить курсор в начало документа. Дайте команду Правка Вставить, чтобы вставить фрагмент из буфера обмена.
8. Установите указатель мыши на начало слова Вольтметр. Нажмите левую кнопку мыши и, не отпуская ее, выделите это слово методом протягивания.
9. Нажмите комбинацию клавиш CTRL+X, переместите текстовый курсор в начало второй строки текста и вставите новый фрагмент из буфера обмена (CTRL+V).
10. Установите текстовый курсор в начало строки, содержащей слова Катушка индуктивности. Дважды нажмите комбинацию SHIFT+CTRL+ВПРАВО и убедитесь, что при каждом нажатии выделенный фрагмент расширяется, охватывая следующее слово. Нажмите комбинацию клавиш SHIFT+ВПРАВО. Мы выделили нужный фрагмент при помощи клавиатурных команд.
11. Нажмите комбинацию клавиш SHIFT+DELETE, переместите текстовой курсор в начало третьей строки текста и вставьте новый фрагмент из буфера обмена с помощью комбинации клавиш SHIFT+INSERT.
12. Используя описанные приемы, завершите формирование списка введенных терминалов в алфавитном порядке.
13. Сохраните созданный документ под именем list.txt.

ЛАБОРАТОРНАЯ РАБОТА № 11

ТЕКСТОВЫЙ РЕДАКТОР WordPad

Загрузите редактор WordPad (Пуск / Программы / Стандартные / WordPad). Сравните окно этого редактора с окном Блокнота. Обратите внимание на ***панели инструментов*** – стандартную (верхняя строка), предназначенную для управления файлами, буфером обмена и некоторых других действий, и ***панель форматирования***. Опробуйте включение и отключение ***линейки***, строки состояния (используйте меню Вид). Сравните команды меню в Блокноте и WordPad.

Выполните в WordPad те же действия, что и в Блокноте, кроме сохранения документа. При переносе и копировании используйте кнопки на панели инструментов.

Измените форматирование подготовленного текста. Выделите фамилию и измените ее шрифт: ***тип шрифта*** – Courier New (Кириллица), ***размер*** – 14, ***начертание*** – полужирный курсив (нажмите кнопки **Ж** и **К**).

Выделите весь текст и скопируйте его дважды на этот же лист ниже. Для каждой копии выберите разное ***выравнивание***: по левому краю, по правому краю, по центру (выделить фрагмент и нажать соответствующую кнопку на панели форматирования). Снимите выделение.

Опробуйте изменение ***границ абзаца*** с помощью ***маркеров***, установленных на линейке (треугольники и квадратик слева и треугольник справа). Найдите маркеры, переместите их с помощью мыши в разные положения вдоль линейки, верните в исходное положение. Для изменения границ первого абзаца выделите его и перетащите левый маркер на два сантиметра вправо: выделенный абзац переместился в соответствии с положением маркера. Снимите выделение, установите курсор в первую строку и нажмите клавишу табуляции **Tab**: установлен ***абзацный отступ*** (“красная строка”). Для отмены ошибочно выполненных действий используйте кнопку ***Отменить*** (найдите ее на панели инструментов с помощью всплывающей подсказки).

Сохраните файл под именем Формат, выбрав тип файла Word для Windows 6.0. Просмотрите свойства файла, обратите внимание на расширение, значок. Выполните Быстрый просмотр.

Самостоятельная работа

В этом упражнении мы создадим иллюстрированный словарь терминов, введенных в файл list.txt.

1. Запустите текстовый процессор WordPad (Пуск Программы Стандартные WordPad).
2. Откройте текстовый файл list.txt.
3. Дайте команду Файл Сохранить как, в списке Тип файла выберите пункт Word для Windows 6.0 и сохраните файл под именем dict.doc.
4. Выделите первое слово документа (Амперметр). На панели форматирования задайте шрифт Arial (Кириллица), размер шрифта – 14 пунктов, выберите полужирное начертание.
5. Нажмите клавишу END, чтобы снять выделение, а затем – клавишу ENTER.
6. Введите краткое описание термина, указанного в предыдущей строке, например так: “прибор для измерения величины электрического тока”. Размножьте введенный текст таким образом, чтобы образовался абзац размером 3–4 строки.
7. Выделите весь только что введенный абзац (можно использовать “тройной щелчок”). На панели форматирования задайте шрифт Times New Roman (Кириллица), размер шрифта – 12 пунктов.
8. На линейке, расположенной ниже панели форматирования, перетащите маркер в виде квадратика на расстояние 1 см (по линейке) вправо. Убедитесь, что весь абзац теперь отображается с отступом от левого края.
9. Снимите выделение и установите курсор в начало первой строки того же самого абзаца. Нажмите клавишу TAB. Убедитесь, что табуляция в

первой строке абзаца может использоваться для создания абзацного отступа.

10. Введите аналогичные краткие описания для последующих терминов создаваемого “словаря” и отформатируйте термины и описания так, как указано в пп. 4–9.
11. Установите курсор в конец описания термина Вольтметр и нажмите клавишу ENTER.
12. Дайте команду Вставку Объект. В диалоговом окне Вставка объекта установите переключатель Создать из файла.
13. Щелкните на кнопке Обзор, разыщите в файловой структуре ранее созданный документ scheme.bmp, щелкните на кнопке Вставить. Щелкните на кнопке ОК.
14. Убедитесь, что созданное изображение схематического обозначения вольтметра вставлено в документ в качестве иллюстрации.
15. Измените масштаб отображения рисунка в документе путем перетаскивания маркеров изменения размера, расположенных на границах объекта.
16. Сохраните текущий документ dict.doc.

ЛАБОРАТОРНАЯ РАБОТА №12

ГРАФИЧЕСКИЙ РЕДАКТОР Paint

Загрузите редактор Paint (Пуск / Программы / Стандартные / Paint). Просмотрите команды меню. Удалите и восстановите ***набор инструментов для рисования***, ***палитру***. Рассмотрите (используя всплывающие подсказки) каждый инструмент. Определите установленный размер листа в точках (Рисунок / Атрибуты), проверьте установку непрозрачного фона и масштаба 100%.

Опробуйте работу каждого инструмента. Используя инструмент Прямоугольник, нарисуйте на листе восемь непересекающихся прямоугольни-

ков (окон) примерно одинакового размера: после щелчка по кнопке Прямоугольник нажмите левую кнопку мыши в левом верхнем углу изображаемого окна, протащите мышь до правого нижнего угла и отпустите кнопку. Повторите эти действия восемь раз, меняя положение углов прямоугольников.

В первом окне опробуйте работу инструмента Карандаш, во втором – Кисти (выбирая разную ширину и срез кисти в панели под набором инструментов и разные цвета на палитре). В третьем окне проведите несколько прямых разной толщины, используя инструмент Линия. Для проведения строго горизонтальных или вертикальных линий, или линий под углом 45 градусов, держите нажатой клавишу **Shift**. В других окнах нарисуйте пересекающиеся прямоугольники (всех трех типов), эллипсы (при нажатой клавише **Shift** - окружности), скругленные прямоугольники, многоугольники (при рисовании многоугольника щелкайте мышью в каждом его углу). В последнем окне опробуйте построение кривых: протащите мышь от начальной точки кривой к конечной и отпустите кнопку; установите указатель мыши в средней части линии и протащите мышь до получения нужной кривизны. Попробуйте построить кривую с двумя точками перегиба.

В различных окнах опробуйте работу Заливки (разными цветами), Распылителя (разных цветов и типов), Ластика для стирания графических элементов (разных размеров), измените масштаб: увеличьте изображение вдвое, в 6 раз, вернитесь к исходному масштабу. Выберите инструмент Надпись, и в двух разных окнах сравните варианты прозрачной и непрозрачной надписей (выбирается под инструментами); измените тип шрифта, размер, начертание с помощью панели атрибутов текста (в меню Вид).

Используя команды меню Рисунок, опробуйте другие возможности графического редактора: растяжение и наклон изображения, отражение, поворот, обращение цветов. Выделите прямоугольную область, например, в виде двух нарисованных окон, с помощью инструмента Выделение; исполь-

зуя контекстное меню скопируйте эту область в буфер, затем вставьте в нижнюю часть листа (после щелчка Вставить копия выделенной области появляется в левом верхнем углу листа, откуда мышью перетаскивается в нужное место). Опробуйте по аналогии работу инструмента Выделение произвольной области. Выделенная область может быть скопирована или перенесена не только на данный лист, но и в любой другой документ (позволяющий встраивание графики, в частности, в документ WordPad).

Просмотрите расположение рисунка на листе бумаги (Файл / Предварительный просмотр). Параметры бумаги могут быть изменены с помощью меню Файл / Макет страницы. Сохраните рисунок на дискете в виде файла с именем Мой рисунок. Просмотрите свойства файла. Обратите внимание на расширение и значок. Выполните Быстрый просмотр, закройте окно. Используя папку Мой компьютер, загрузите рисунок двойным щелчком по значку. Закройте редактор.

Самостоятельная работа

В этом упражнении мы создадим условное обозначение вольтметра, принятое на электрических схемах.

1. Запустите графический редактор Paint (Пуск Программы Стандартные Paint).

2. Убедитесь, что на палитре задан черный цвет в качестве основного и белый – в качестве фонового.

3. Дайте команду Рисунок Атрибуты, в диалоговом окне Атрибуты задайте ширину рисунка, равную 300 точек и высоту – 200 точек. Щелкните на кнопке ОК.

4. Выберите инструмент Эллипс и в палитре настройке инструмента укажите вариант Без заполнения.

5. Накажите и удерживайте клавишу SHIFT. Методом протягивания нарисуйте окружность в центральной части области рисунка. Диаметр

окружности должен составлять около половины высоты рисунка. Отпустите клавишу SHIFT.

6.Выберите инструмент Линия. В палитре настройки инструмента выберите вариант толщины линии (второй сверху).

7.Нажмите и удерживайте клавишу SHIFT. Методом протягивания нарисуйте небольшой горизонтальный отрезок прямой в стороне от окружности. Отпустите клавишу SHIFT.

8.Выберите инструмент Выделение. В палитре настройки инструмента выберите режим с прозрачным фоном.

9.Методом протягивания выделите прямоугольный фрагмент, охватывающий нарисованный отрезок прямой, но не затрагивающий окружность. Комбинацией клавиш CTRL+X поместите его в буфер обмена.

10.Вставьте отрезок прямой на рисунок комбинацией клавиш CTRL+V. Обратите внимание, что выделение при этом сохраняется.

11.Переместите выделенный фрагмент так, чтобы отрезок прямой примыкал к окружность слева. Обратите внимание на то, что фоновая часть фрагмента не перекрывает окружность.

12.Повторите операции, описанные в пп. 10-11, чтобы создать отрезок прямой, примыкающий к окружности справа.

13.Выберите инструмент Текст.

14.Методом протягивания создайте область ввода текста внутри окружности. Введите символ “V”. С помощью панели Шрифты задайте подходящий размер и начертание шрифта.

15.Методом перетаскивания за границу области ввода текста поместите букву “V” в центре окружности.

16.Щелкните вне области ввода текста, чтобы превратить текст в часть рисунка.

17.Сохраните созданное изображение под именем scheme.bmp.

ЛАБОРАТОРНАЯ РАБОТА №13

ТЕКСТОВЫЙ ПРОЦЕССОР MS WORD

Цель работы: получить практические навыки по выполнению многоколоночной верстки и оформлению текстовых документов со вставками. Научиться оформлять большие документы.

Должны знать: основные команды строки меню, основные команды меню Вставка, Формат.

Должны уметь: правильно вводить тексты, форматировать текст, изменять стили заголовков, добавлять содержание в документ, применять автоматическую нумерацию страниц.

Задание №1.

Научиться форматировать тексты

- Установить параметры страницы верхнее – 2 см, нижнее – 2 см, левое – 2,5 см, правое – 1 см.
- Набрать представленный ниже текст.

Категории швейных предприятий сервиса

Дома моды и ателье могут быть различных разрядов в зависимости от заданного качества услуг и обслуживания, новизны, разнообразия и сложности выполняемых фасонов изделий, создания комфортных условий в салоне для заказчиков.

Дома моды категории «люкс» предназначены для изготовления модных индивидуальных изделий любой сложности из всех видов материалов, включая изделия перспективной моды, в ансамбле, вечерних, эстрадных и сценических туалетов, уникальных авторских изделий. В домах моделей «люкс» заказчику предоставляют: услуги художника-модельера, услуги информационной службы. Качество услуг и обслуживания в домах моделей «люкс» должно быть на самом высоком уровне, с использование всех передовых технологий и методов, в т.ч. высоких технологий.

Дома моды и ателье высшего разряда предназначены для изготовления моделей индивидуального решения и оригинальных моделей любой сложности современной и перспективной моды, а также для выполнения высокохудожественного ремонта изделий. Прием заказов и предоставляемые услуги заказчику осуществляются аналогично категории «люкс».

Дома моды и ателье первого разряда предназначены для изготовления изделий любой сложности, а также для обновления и ремонта изделий, могут выполняться заказы на изготовление одежды по образцам, пошив малых партий одежды. Заказы принимают по эскизам художника-модельера или заказчика, используют альбомы, каталоги и журналы мод, представленные в выставочном зале образцы.

- Применить шрифт Times New Roman, 14 размер, выровнять по ширине, использовать автоматическую расстановку переносов. Установить отступ первой строки 1,27 см, междустрочный интервал 1,2 см;
- Текст разбить на три колонки. Поставить разделительную черту;
- Установить различную ширину колонок 1 – 7 см;2 – 4 см; 3 – 4,25 см.
- Первый абзац текста, используя вставку надписи, растяните на две колонки;
- Заголовок текста оформите, используя возможности объекта Word Art;
- Вставьте в текст подходящий рисунок, используя параметры обтекания рисунка текстом **вокруг рамки** рисунка. Рисунок разместите в центральной части текста.
- Сохраните документ под именем ***Ателье и Дома Мод***.
- Сравните полученный результат с образцом, приведенным в ПРИЛОЖЕНИИ 1.

Задание №2.

Научиться оформлять большие документы.

Согласно заданию, студенты создают новый файл в текстовом редакторе, в который добавляют файл с именем **текстовое задание** из папки *Мои документы***;** затем выполняют все пункты заданий к практической работе.

•Запустите программу MS Word.

•Создайте новый документ, сохраните его под именем **Большой документ** в папку **Группа 287.**

•Установите параметры страницы для документа:

1. верхнее – 1,5 см
2. нижнее – 1,5 см
3. левое – 2,5 см
4. правое – 1,5 см

Установите флажок **зеркальные поля –** для возможности печати документа с двух сторон листа.

•Вставьте в документ текст из файла **текстовое задание**, расположенного в папке **ТКШИ.**

•В данном документе проверьте правописание, если есть ошибки, то исправьте их.

•Ко всему документу примените шрифт Arial, 12 размер; установите междустрочный интервал – полуторный, текст – по ширине страницы, расставьте переносы слов во всем документе, начало каждого абзаца – с красой строки.

•Добавьте нумерацию страниц: внизу, от центра.

•Измените стиль текста, выделенного красным цветом – **Заголовок 1** (по центру, Arial, 20пт, полужирный, все прописные.

•Для текста, выделенного желтым цветом, примените ***Заголовок 2*** (по левому краю, Times New Roman, 18 пт, полужирный, все прописные)

•Текст, выделенный голубым цветом, сделайте нумерованным списком.

•Вставьте в текст формулу, используя команду меню Вставка – Объект – MS Equation3.0. Пользуясь инструментами этого приложения, постройте формулу так, как она изображена в ПРИЛОЖЕНИИ 2.

•Для слов где S_1 – норма площади на одного работающего, выделенных зеленым цветом, в конце добавьте сноску обычную, в сноске – *Нормы площади, на одного работающего, S_1, рекомендованы ЦОТШЛ*

•Вставьте таблицу, состоящую из 2 столбцов и 15 строк.

•Список литературы разместите на новой странице. (Меню **Вставка** – Разрыв – С новой страницы)

Самостоятельная работа

1.Оформите титульный лист для документа, в котором написан текст, создайте декоративную рамку на титульном листе. Макет титульного листа представлен на рис.1.

2.Добавьте картинку из коллекции ClipArt на титульный лист. Образец титульного листа представлен в ПРИЛОЖЕНИИ 2.

3.Добавьте содержание в конце документа.

4.Покажите выполненную работу преподавателю.

5.Сохраните документ на дискете 3,5 А, компакт-диске или на носителе флэш-памяти.

Министерство образования российской федерации
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Реферат на тему:
«Швейные предприятия сервиса»

Выполнил: Петров В.Н.
Проверил: Евгеньева А.М.

Благовещенск
2005г.

Рис. 1. Титульный лист большого документа

ЛАБОРАТОРНАЯ РАБОТА №14

ТЕХНОЛОГИИ ОБРАБОТКИ ГРАФИЧЕСКОЙ ИНФОРМАЦИИ

Цель работы: научиться пользоваться инструментами панели **Рисование** MS Word, а также пользоваться графическим редактором MS Visio.

Должны знать: С помощью инструментов панели **Рисование** MS Word научиться создавать автофигуры, добавлять текст в автофигуру, группировка объектов, настройка объема и тени; и дополнительно: установление динамических связей между графическими объектами MS Visio.

Должны уметь: правильно устанавливать размеры автофигуры, добавлять текст в автофигуру, группировать несколько автофигур, изменять форматы созданных объектов.

1.Создайте различные схемы, графики и рисунки с помощью инструментов панели Рисование MS Word и графического редактора MS Visio.

В ПРИЛОЖЕНИИ 5 приведены примеры схем, которые в данной лабораторной работе необходимо начертить с помощью инструментария **Рисование** MS Word и MS Visio. В графическом редакторе MS Visio следует начертить схемы 1, 3 и 4. ВС помощью панели Рисование MS Word следует нарисовать все схемы. Содержание текстовых надписей может быть произвольным.

Приемы работы, панели инструментов и основные команды по работе с графическим редактором MS Visio приводятся в ПРИЛОЖЕНИИ 4.

2.В начале нарисуйте схему 1. Она состоит из нескольких объектов, соединенных между собой. Шрифт внутри каждого прямоугольника выберете 16пунктов. Нарисованные объекты сгруппируйте.

3.Схема 2 позволит научиться создавать объекты с заливкой и с тенью. Выберите шрифт надписей 14 пунктов, курсивный.

3. Схему 3 нарисовать и в инструментарии MS Word и с помощью графического редактора MS Visio. Обратите внимание на объект **Линия**, который может иметь разные форматы (толщина, тип, цвет, наличие стрелок на

концах). Обязательно сделайте привязки прямоугольников и других фигур ко всем примыкающим к ним линиям.

4. Схема 4 позволяет изобразить пример наиболее часто встречающихся при проектировании предприятий схем. Это схема производственного состава ателье. Нарисуйте ее также, как и предыдущие схемы в Word и MS Visio. Сравните возможности перекомпоновки объектов схемы по содержанию и объему при рисовании в Word и при рисовании в MS Visio

Самостоятельная работа

1. Выполните с помощью панели инструментов MS Word рисунок поздравительной открытки (Рис.2).

При выполнении работы рекомендуется использовать также приложение ***MS Art*** из меню **Вставка/ Объект**. Текст содержания открытки может быть произвольным. Количество объектов в рисунке не менее пяти. При добавлении новых объектов не забудьте про их группировку и порядок размещения – на передний план, задний план и т.д. -

2. Сохранить полученные документы, скопируйте на дискете 3,5 А, компакт-диске или на носителе флэш-памяти.

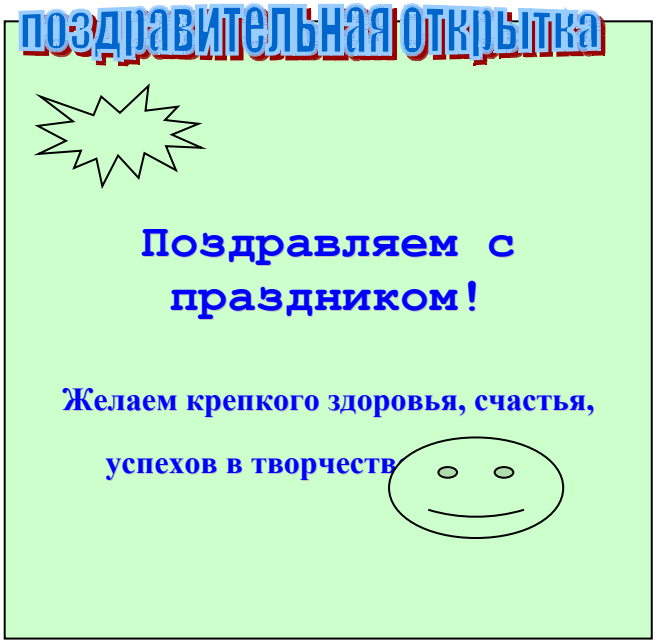


Рис.2. Макет поздравительной открытки

ЛАБОРАТОРНАЯ РАБОТА №15

ТЕХНОЛОГИЯ РАСПОЗНОВАНИЯ ИНФОРМАЦИИ. РАБОТА СО СКАНЕРОМ И ПРИНТЕРОМ.

Цель работы: получить практические навыки по работе со сканером, программой оптического распознавания текста и оформлению текстовых документов, импортированных из других приложений в частности как распознанный текст со вставками. Научиться рационально выбирать параметры сканирования текстовой и графической информации.

Должны знать: основные команды программы по работе со сканером, основные команды программы Abby Fine Reader .

Должны уметь: правильно включать и выключать сканер, работать с мастером распознавания бумажных документов; включать принтер, проверять наличие картриджа и степень его заполнение чернилами; выбирать рациональные опции печати

Методические указания

- Программы для оптического распознавания текстов предназначены для распознавания бумажных документов всех видов и преобразования в электронный вид в форме информации заданного вида (текст, картинка, таблица). Как правило, профессиональные версии такого рода программ - это дорогостоящие программные продукты.
- В данной лабораторной работе предполагается использовать одно из наиболее удобных в работе и популярных приложений - ABBYY FINEREADER. Версия 7.0 HOME EDITION была выпущена В 2005 году. Боле подробно о работе с этой программой, о шагах мастера сканирования и распознавания бумажных документов, о горячих клавишах изложено в ПРИЛОЖЕНИИ 7.

Задание №1. **Отсканировать документ, состоящий из двух страниц**.

Бумажный документ, подлежащий сканированию, состоит из двух страниц с текстом и вставками в виде рисунков, таблицы, формулы. Пример данного документа приведен в Приложении 8.

- Включить сканер, дать ему прогреться.
- Открыть крышку планшета и положить первый лист документа лицевой стороной вниз.
- Запустить программу Abby Fine Reader
- Выполнить первое действие Мастера: Сканировать
- В появившемся диалоговом окне мастера сканирования (программа-приложение к сканеру) выбрать параметры сканирования:

а)Плотность 300 или 600 точек на дюйм

б)Режим – оттенки серого (если рисунок должен остаться цветным, можно выбрать режим RGB). Но следует помнить, что цветная картинка займет места во много раз больше, чем черно-белая.

- Сделать предпросмотр файла, (команда Сканировать), на полученном в окне предпросмотра рисунка убедиться, что страница положена без излишнего наклона, на сканере формата А3 выбрать область сканирования, ограниченную лежащей на планшете страницей формата А4. Для этого область сканирования выделить мышкой.
- Отсканировать страницу окончательно, как файл с сохранением (команда Сканировать/Сохранить). В появившемся окне диалога подтвердить сохранение файла после сканирования в формате *.jpg или *.tiff , выбрав ему имя Лист1.

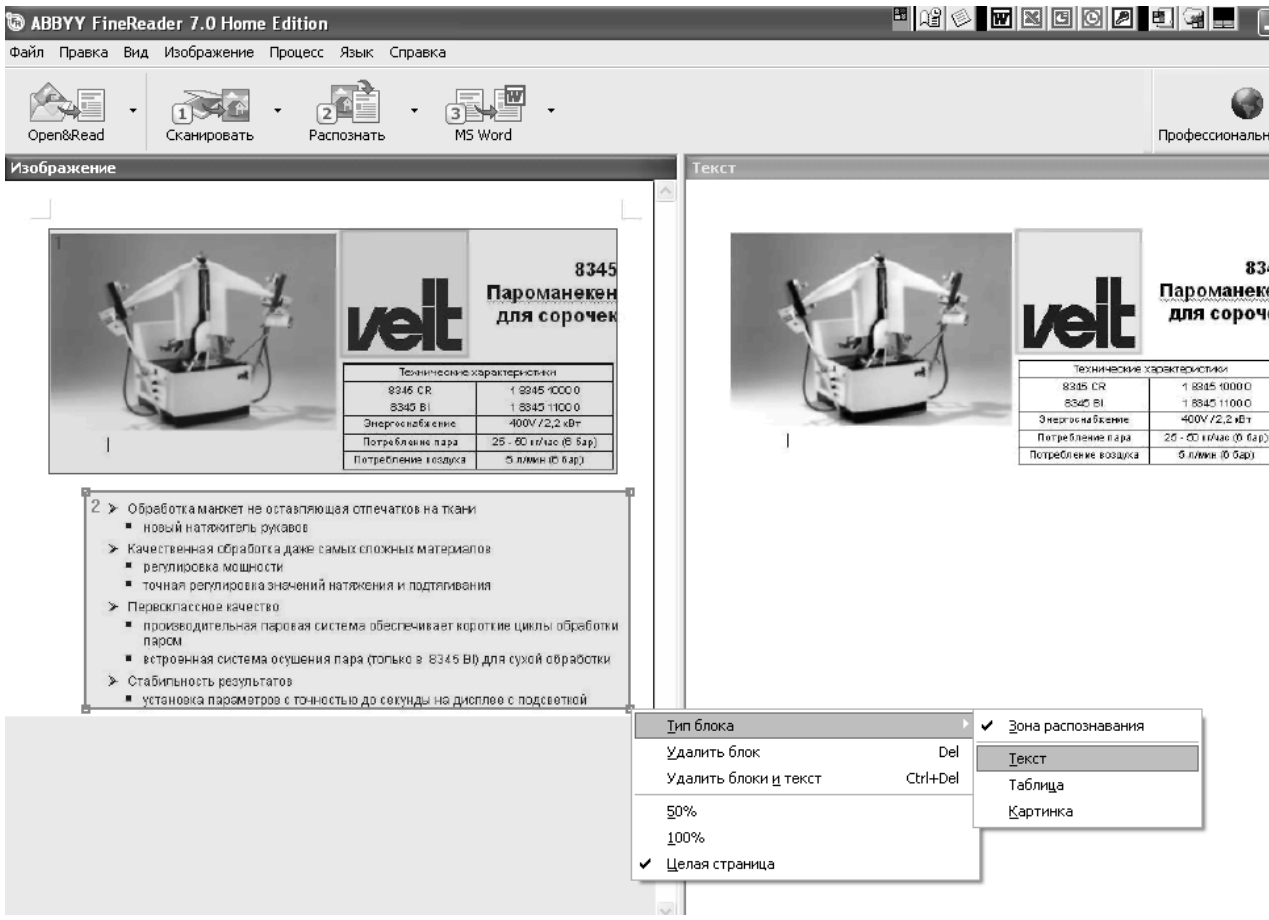


Рис. 3 Пример работы Мастера в приложении Abby Fine Reader.

- Выполнить второе действие Мастера приложения Abby Fine Reader: *Разбить на блоки* (Сегментировать) документ. Всего на первой странице документа 5 сегментов (На рис.3 показана только часть листа для распознавания, но присутствуют все пять элементов). Следует отметить, что 7-я версия Abby Fine Reader автоматически разбивает документ на блоки, не фиксируя этот этап отдельно.
- Если программа не распознала элемент 4 как таблицу, то это надо сделать вручную. Правой кнопкой мыши вызвать контекстное меню и там выбрать команду **Распознать как / Таблица**.
- После сегментирования запустить третий шаг Мастера: *Распознать*. В результате с правой стороны окна программы заполняется окно с распознанным текстом.
- Сделать проверку полученного документа на наличие орфографических ошибок.

- Аналогичным образом распознать вторую страницу бумажного документа (сканировать, разбить на блоки, распознать).
- Импортировать полученный документ в приложение MS Word, где сохранить его под именем ПВМ.rtf. Шаблон *. Rtf будет присвоен документу автоматически при импорте из приложения Abby Fine Reader.

Самостоятельная работа

- Открыть сохраненный документ ПВМ.rtf и сохранить его в виде *.doc. В качестве имени студенты могут выбрать любое название, например, свою фамилию.
- Создать в документе *колонтитулы*: верхний с именем и фамилией, а так же названием лабораторной работы, нижний с номером страниц и количеством страниц в документе (2 страницы).
- В меню **Файл** выбрать команду **Печать** и изучить Опции печати.
- Зайти в Меню Свойства принтера, выбрать Качество печати Нормальное. Проверить наличие чернил в черном и цветных картриджах. Для этого в Свойствах принтера, в меню сервис выбрать Панель инструментов принтера, где выбрать закладку *Предполагаемый уровень чернил*.
- Распечатать документ в виде одной копии документа так, чтобы на листе формата А4 поместилось 2 страницы. Для этого в диалоговом окне *Опции Печати* выбрать (справа) *Количество страниц на листе* – 2.
- Сдать полученные бумажные документы на подпись преподавателю, полученные файлы переписать на дискету 3,5 А, компакт-диск или на носитель флэш-памяти.

ЛАБОРАТОРНАЯ РАБОТА №16

СОЗДАНИЕ И НАСТРОЙКА Power Point.

Цель работы: ознакомиться с возможностями программы создания презентаций, научиться создавать, сохранять и открывать презентацию; вводить текст на слайд, форматировать и редактировать его.

Вопросы для подготовки к работе:

1) Сформулируйте назначение и основные возможности Power Point.

2) Приведите примеры области применения программы, в том числе в сфере сервиса. Каковы психологические проблемы использования рекламных презентаций?

3) Продумайте тему будущей презентации. Студенты дневного отделения могут выполнить презентацию по одной из лабораторных работ по дисциплине «Технологические процессы в сервисе» или любой другой. Студенты заочной формы обучения могут создать демо-ролик (рекламный ролик), представляющий швейное предприятие сервиса, на котором они работают.

Методические указания:

1.Запустить программу Power Point.

2.Выбрать создание **новой** презентации. Из образцов макетов выбрать **пустой слайд**.

3.Выбрать оформление слайда в соответствии с темой представленным на образце.

1. Выбрать фон оформления, используя команду **Оформления слайда** из меню **Формат**. Необходимо выбрать шаблон оформления, готовую цветовую схему или изменить цветовую схему по своему выбору.
2. Заполните поля слайда объектами (заголовком надписями и рисунками), используя команды из меню **Вставка.** Разместите объекты на слайде так, как они представлены на рисунке.

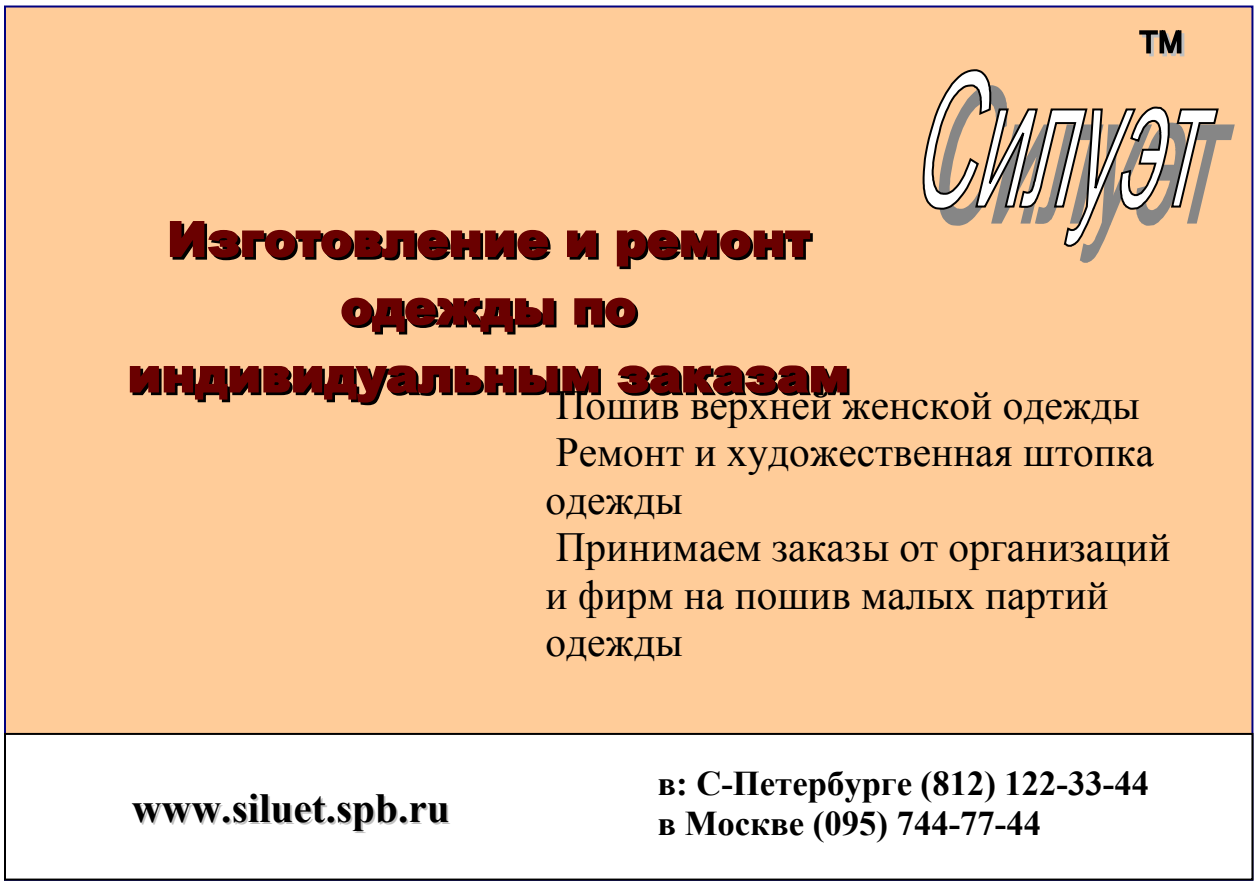


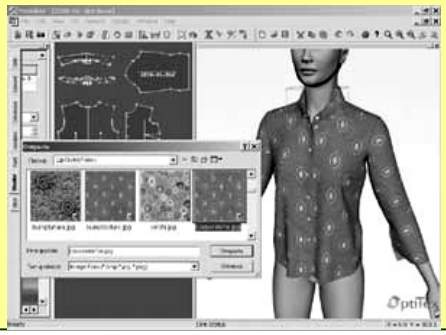
Рис. 4. Схема первого слайда презентации

4.Добавьте в вашу презентацию еще минимум 4 новых слайда по данной тематике. Если выполняется презентация с целью представления швейного предприятия, то вся информация должна иметь целостность и вызывать интерес к услугам швейного предприятия сервиса. Для добавления нового слайда воспользуйтесь командой **Вставка слайда** из меню **Вставка**.

Образцы слайдов по теме презентации, связанной с рекламой швейного предприятия представлены на следующей странице.

Только у нас в салоне!

Компьютерный показ моделей и трехмерная визуализация будущей модели на трехмерном манекене



Художник модельер

Силуэт

Проводит консультации заказчикам
оказывает помощь

Индивидуальный подход к каждому клиенту
Выбор модели и материалов, цветовой гаммы
Каталог эксклюзивной фурнитуры – только для клиентов нашего салона
Гибкая система скидок для постоянных клиентов
Ремонт одежды и художественная штопка
Прокат элитной одежды и аксессуаров

Наш адрес: С-Петербург, Невский пр., 15-А

В нашем салоне

Силуэт

Только для ВАС

Пошив одежды по индивидуальному решению
Ремонт одежды и художественная штопка
Гибкая система скидок для постоянных клиентов
Прокат элитной одежды и аксессуаров

Наш адрес: С-Петербург, Невский пр., 15-А

Рис.5. Примеры слайдов в тематической презентации по швейному предприятию сервиса

5.Настройте анимацию, выбрав из меню **Показ слайда** команду **Настройка анимации**. Для каждого объекта на слайде включите анимацию, применив опцию Автоматическое включение через 1-5 секунд.

6.Добавьте звуковое сопровождение к анимированным объектам слайдов. Достаточно сделать это лишь на некоторых из них, т.к. чрезмерная перегрузка звуком и анимацией может привести к антирекламному эффекту ролика.

7.В меню **Смена слайдов** выберете тип смены слайда и автоматический режим сменности слайдов.

8.Проверьте полученный результат с помощью команды **Начать показ**.

9. Сохраните презентацию в виде двух файлов: файл презентации (шаблон файла *.ppt) и как демо-ролик (*.pps.)

ЛАБОРАТОРНАЯ РАБОТА №17

СОЗДАНИЕ ТАБЛИЦ И ВЫПОЛНЕНИЕ РАСЧЕТОВ В ПРОГРАММЕ MS Excel

Цель работы: Научиться производить вычисления в программе MS Excel и строить круговые диаграммы, ознакомиться с методами сортировки данных. Научиться создавать формулы в MS Excel. Научиться добавлять примечания и фильтровать данные

Должны знать: основные команды меню **Формат**, **Вставка,** команды меню **Сортировка** из меню **Данные,** правила ввода формул и создания диаграмм.

Должны уметь: вводить данные в ячейки таблицы MS Excel, вставлять рисунки в документ, вводить формулы и производить вычисления с помощью формул, используя мастер функций, форматировать таблицу**,** в программе MS Excel, строить диаграммы, оформлять их, изменять вид и параметры диаграммы.

Методические указания

Электронные таблицы, в частности **MS Excel,** предоставляют комплексные средства для хранения различных типов данных и их обработки. В некоторой степени они аналогичны системам управления базами данных, но основной акцент смещен не на хранение массивов данных и обеспечение к ним доступа, а на преобразование этих данных и их обработку. Рассмотрим подробнее общее принципы построения электронных таблиц.

Документ Excel называется рабочей книгой. Рабочая книга представляет собой набор рабочих листов, каждый из которых имеет табличную структуру и может содержать одну или несколько таблиц. В окне документа в программе Excel отображается только текущий рабочий лист, с которым и ведется работа. Каждый рабочий лист имеет название, которое отображается на ярлычке листа, отображаемом в его нижней части. С помощью ярлычков можно переключаться к другим рабочим листам, входящим в ту же самую рабочую книгу. Чтобы переименовать рабочий лист, надо дважды щелкнуть на его ярлычке.

Рабочий лист состоит из строк и столбцов Столбцы озаглавлены прописными латинскими буквами и далее двухбуквенными комбинациями. Всего рабочий лист может содержать до256 столбцов пронумерованных от A до IV. Строки последовательно нумеруются цифрами от 1 до 65 536 (максимально допустимый номер строки).

Ячейки и их адресация. На пересечении столбцов и строк образуются ячейки таблицы Они являются минимальными элементами для хранения данных Обозначение отдельной ячейки сочетает в себе номера столбца и строки (в этом порядке), на пересечении которых она расположена например A1 или DE234. Обозначение ячейки (ее номер) выполняет функции ее адреса. Адреса ячеек используются при записи формул, определяющих взаимосвязь между значениями, расположенными в разных ячейках

Итоговые вычисления в Excel предполагают получение числовых характеристик, описывающих определенный набор данных в целом. Например, возможно вычисление суммы значений входящих в набор, среднего значения

и других статистических характеристик количества или доли элементов набора удовлетворяющих определенных условиям.

Проведение итоговых вычислений в программе Excel выполняется при помощи встроенных функций. При этом программа может задать параметры функции автоматически.

В программе Excel возможно построение *диаграмм и графиков.* Термин диаграмма используется для обозначения всех видов графического представления числовых данных. Построение графического изображения производится на основе ряда данных. Так называют группу ячеек с данными в пределах отдельной строки или столбца. На одной диаграмме можно отображать несколько рядов данных. Диаграмма представляет собой вставной объект, внедренный на один из листов.

Такой широкий спектр возможностей открывает всевозможные области применения электронных таблиц практически во всех отраслях, в частности в швейной отрасли.

На швейных предприятиях сервиса технологические операции по обработке изделий напрямую связаны с нормативами времени. Все нормативы могут быть представлены в виде большой таблицы (совокупности таблиц) с данными. Как правило технологу швейного производства требуется составить технологическую последовательность на изделие и сосчитать суммарную затрату времени на его изготовление. В том числе требуется проанализировать состав отобранных в последовательность операций по видам работ, т.к. в дальнейшем придется разделять все операции по видам работ для организации специализированных рабочих мест.

Нормативы времени, заключенные в большие массивы данных в виде таблиц, представляют собой емкий источник информации. Это не только цифры, это и иды работ, и оборудование, это сами неделимые операции, среди которых находятся нужные для составления конкретной технологической последовательности. Технолог отбирает нудные операции, а рутинные расчеты выполняет программа.

Рекомендации по работе технолога в MS Excel для составлении технологической последовательности и анализа методов обработки приводится в ПРИЛОЖЕНИИ 5.

ЗАДАНИЕ. Выбрать необходимые операции их таблицы, отфильтровать их и проанализировать состав.

Запустите программу MS Excel.

Откройте файл **ТНВ Брюки.xls** из папки **Группа 287**

- Просмотрите таблицу, содержащую сведения об операциях на изготовление мужских и женских брюк.
- Создать список по содержимому первого листа **Разделы,** используя команду **Форма** из меню Данные
- Перейти на лист **НормативБрюки.** Изучить поля таблицы
- Для фиксации верхней строки использовать команду **Закрепить области** из меню **Окно**; прокрутить весь массив данных до конца.
- Изменить заголовок таблицы. Для заголовка таблицы применить шрифт Arial, 18 размер;
- Отформатировать ширину ячеек таблицы так, чтобы все столбцы были видны на экране, используя команды меню **Формат**. Отформатировать таблицу, используя цвет шрифта, заливку и границу с помощью команд окна диалога **Формат ячеек.** Выровнять содержимое ячеек по значению и отображение с переносом по словам. Отформатировать высоту строк в меню **Формат строки / Автоподбор высоты.**
- По заданию преподавателя занести в таблицу *новую информацию*: новую технологическую операцию в имеющийся список технологических операций.
- Добавьте примечание *для новой операции*. Добавить примечание с помощью команды Примечание из меню Вставка.

•Отфильтровать данные, используя команду Фильтр (Автофильтр) из меню Данные. Задание: из имеющейся базы данных технологических операций отобразите на экране операции, относящиеся к разделу № 12

•В открытый файл добавить новый лист и присвоить ему имя Рабочий лист.

•На полученном листе создать таблицу и скопировать в нее данные, отобранные на листе норматив

•Вставить новый столбец перед столбцом Оборудование. Назвать столбец Новый норматив времени

•Ввести формулу в первую ячейку нового столбца для получения результата новой нормы времени с учетом повышающего коэффициента к=1,25

•Скопировать эту формулу на остальные ячейки этого столбца. Просуммировать результат по всему диапазону ячеек. С помощью кнопок на панели инструментов уменьшить разрядность вычисленных значений и установите точность – 2 десятичных знака после запятой.

•Отфильтруйте данные в столбце по виду работ: М. Для выбранного диапазона в столбце Новый норматив измените формулу, используя коэффициент 1,1. Отобранный диапазон скопируйте.

•Вставьте новый лист **Машинные работы** и вставьте скопированные данные. Отформатируйте новую таблицу так, как на листе **Нормы времени.**

•Проделайте аналогичную фильтрацию данных по видам работ Р, У и СМ. Для данных с видом работ Ручные (Р) измените коэффициент к=1,35. Для данных по виду работ Утюжильные (У) измените формулу в новом столбце норматива: = Первоначальное время*1,2–0,25 (мин.) Каждый отобранный диапазон вставьте на новый лист с соответствующим названием.

•Создайте новый лист **Сводный.** В нем таблицу, состоящую из трех столбцов: Вид работ, Норматив времени и Новый норматив.

•На листе **Сводный** выделите в таблице столбцы со старыми и новыми нормами времени (Столбцы Норма времени и Новый норматив времени)

•Постройте круговую диаграмму по данным выделенных столбцов, разместив диаграмму на том же самом листе, используя команду Диаграмма из меню Вставка.

•Постройте различные гистограммы по данным столбцов Норма времени и Новый норматив времени, используя все 4 шага построения диаграмм. Для этого Выделив в таблице нужный для построения диапазон ячеек, построить с помощью Мастера диаграмм график функции, подобный тому, какой изображён в Приложении 5.

•Отформатировать область графика по собственному усмотрению, используя различные цвета заливки, границ, размеры шрифта.

ЛАБОРАТОРНАЯ РАБОТА №18

РАБОТА С БАЗОЙ ДАННЫХ СУБД ACCESS

Цель работы: Закрепить навыки по заполнению и редактированию таблиц базы данных научиться создавать простые формы.

При выполнении работы студенты

Должны знать: принципы построения таблиц с данными, задание полей таблицы; отличительные особенности запросов на выборку, запросов на создание таблиц и запросов на обновление; особенности конструирования форм и отчетов, знать назначение кнопок Панели элементов и использование их при создании отчетов и форм.

Должны уметь: создавать таблицы с данными, запросы на выборку, запросы на создание таблиц и перекрестные запросы; разрабатывать формы, в том числе главную форму, подготавливать отчеты для печати; При создании отчетов и форм вводить тексты заголовков, добавлять поля из списков, изменять свойства полей в зависимости от их назначения

Содержание работы

- Создание базы данных, состоящей из нескольких таблиц Подстановка данных путем импортирования данных имеющихся таблиц
- Создание запросов
- Создание простых форм
- Создание отчетов
- Создание макросов и главной формы

Методические указания

Основные понятия, применяемые при работе с базами данных, приводятся в Приложении 6. В Приложении 7 изложен порядок работы с СУБД ACCESS при создании таблиц, запросов, отчетов и форм. Приводятся наглядные примеры по работе с реляционной базой данных.

В данной лабораторной работе студенты должны разработать свою базу данных, удобную для работы, поиска и отбора информации. Запросы и формы должны быть разработаны в соответствии с требованиями, изложенными в Приложении 7. Для вывода полученных результатов отбора на печать необходимо создать отчеты.

Порядок проведения работы следующий.

ЗАДАНИЕ по созданию базы данных содержит 4 этапа:

1)Создание таблиц. Импорт данных.

2) Создание запросов.

3) Создание отчетов.

4) Создание форм. Создание макросов.

ЗАДАНИЕ 1.Создание таблиц. Импорт данных.

1. Создайте в Вашей папке базу данных с названием **Нормы времени**. В ней будет три таблицы: **Норматив**, **Разделы** и **Оборудование**.
2. В разделе Таблицы в режиме конструктора создайте структуру таблицы Норматив. Для этого введите следующие поля:

Код операции, тип поля – счетчик;

Номер операции по сборнику, тип поля – текстовый, размер поля - 20 знаков;

Содержание операции, тип поля – поле МЕМО;

Вид работ; тип поля – текстовый, размер поля - 3 знака;

Разряд; тип поля – числовой

Норматив 1; тип поля – числовой, в графе Описание введите Норма времени на мужское пальто;

Норматив 2; тип поля – числовой, в графе Описание введите Норма времени на женское пальто;

Норматив 3; тип поля – числовой, в графе Описание введите Норма времени на пиджак;

Норматив 4; тип поля – числовой, в графе Описание введите Норма времени на жакет;

Норматив 5; тип поля – числовой, в графе Описание введите Норма времени на мужские брюки;

Норматив 6; тип поля – числовой, в графе Описание введите Норма времени на женские брюки;

Оборудование; тип поля – текстовый, размер поля - 100 знаков;

Сделайте поле *Код операции* ключевым. Сохраните структуру данных.

Конструктор таблицы **Норматив** должен выглядеть так, как на рис. 6

3. Выполните импорт данных в таблицу **Норматив**. Для этого В меню **Файл** выберите *Внешние данные* - *импорт*. В открывшемся диалоговом окне выберите тип файла – документ Excel. В рабочей папке **Группа 2307** выберите файл **ТНВ** (см. лабораторная работа №5). Проведите все операции по импортированию данных из листа ***ПальтоПиджакЖакет*** , следуя указаниям мастера. Следует иметь в виду, что первая строка импортируемой таблицы является ее шапкой. Сохраните результат импорта в имеющуюся таблицу **Норматив**.

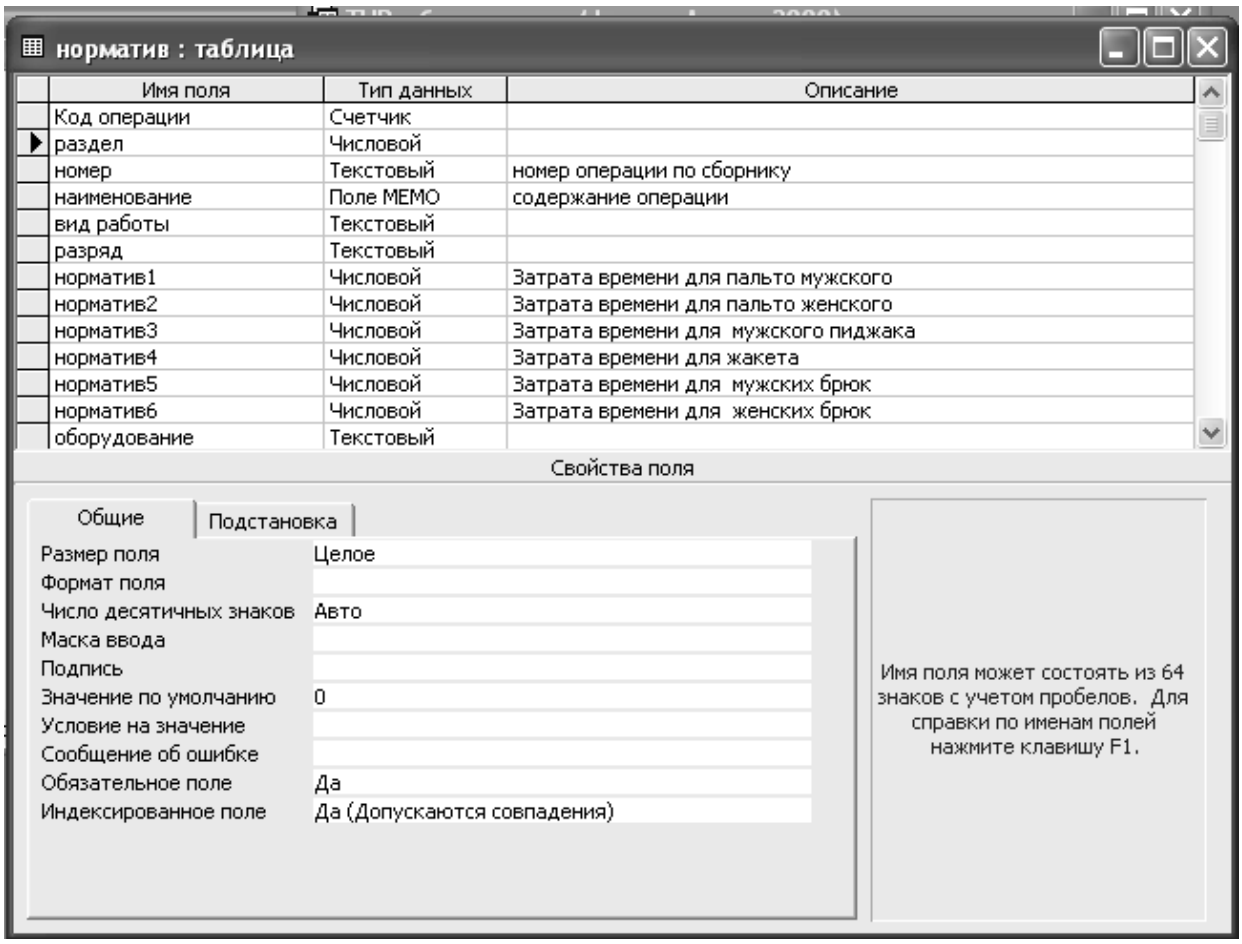


Рис.6. Таблица Норматив в режиме Конструктора.

4. Откройте таблицу **Норматив** в режиме ввода данных **Вид**; установите ширину всех полей таблицы, достаточную, для просмотра данных. Закройте таблицу. На вопрос о сохранении изменения макета таблицы ответьте утвердительно.
5. В разделе **Таблицы** в режиме конструктора создайте структуру таблицы **Разделы** с полями "Номер раздела" и "Наименование раздела" и типом данных "Текстовый". Поле **Номер раздела** сделайте **ключевым**. Размер поля "Наименование раздела" - 100 знаков.
6. Выполните импорт данных в таблицу **Разделы**. Проведите все операции по импортированию данных из листа ***Разделы***, следуя указаниям мастера. Следует иметь в виду, что первая строка импортируемой таблицы является ее шапкой. Сохраните результат импорта в имеющуюся таблицу **Разделы**.

7. Откройте таблицу **Разделы** в режиме ввода данных **Вид**; установите ширину полей таблицы, достаточную, для просмотра данных. Закройте таблицу. На вопрос о сохранении изменения макета таблицы ответьте утвердительно.
8. В разделе **Таблицы** в режиме конструктора создайте структуру новой таблицы **Оборудование** с полями "Код оборудования" (тип данных – счетчик), "Наименование оборудование" (тип данных – текстовый, размер 100 знаков) и поле **Вид работ** (тип поля – текстовый, размер 3знака). Поле **Код оборудования** сделайте **ключевым**.
9. Откройте таблицу **Оборудование** в режиме ввода данных **Вид**; установите ширину полей таблицы, достаточную, для просмотра данных. Введите данные в таблицу, используя данные из электронной таблицы **THB**, лист ***ПальтоПиджакЖакет***. Весь список оборудования можно просмотреть через фильтр данных (см. лабораторная работа №5). Введите несколько видов оборудования, как показано на рис.6, но не менее 15. Закройте таблицу **Оборудование**. На вопрос о сохранении изменения макета таблицы ответьте утвердительно.

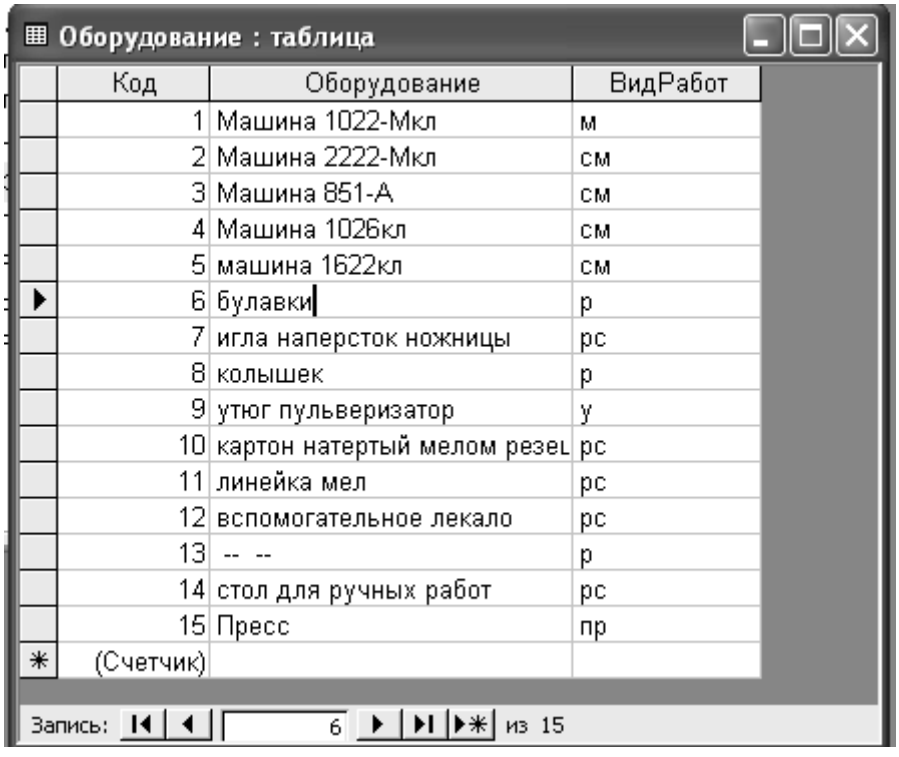


Рис.6. Таблица Оборудование в режиме ввода данных

10. Измените макет таблицы Норматив, создав там поле со списком. Откройте таблицу Норматив в режиме конструктора. Для ввода данных о оборудовании встаньте на поле "Оборудование", перейдите на закладку "Подстановка" и выберите тип элемента управления "**Поле со списком**". Укажите тип источника строк: **Таблица или запрос**. Для выбора источника подстановки щелкните в разделе **Источник строк** и выберите таблицу **Оборудование** (рис.7). Закройте окно. На вопрос о подтверждении обновления свойств ответьте утвердительно.

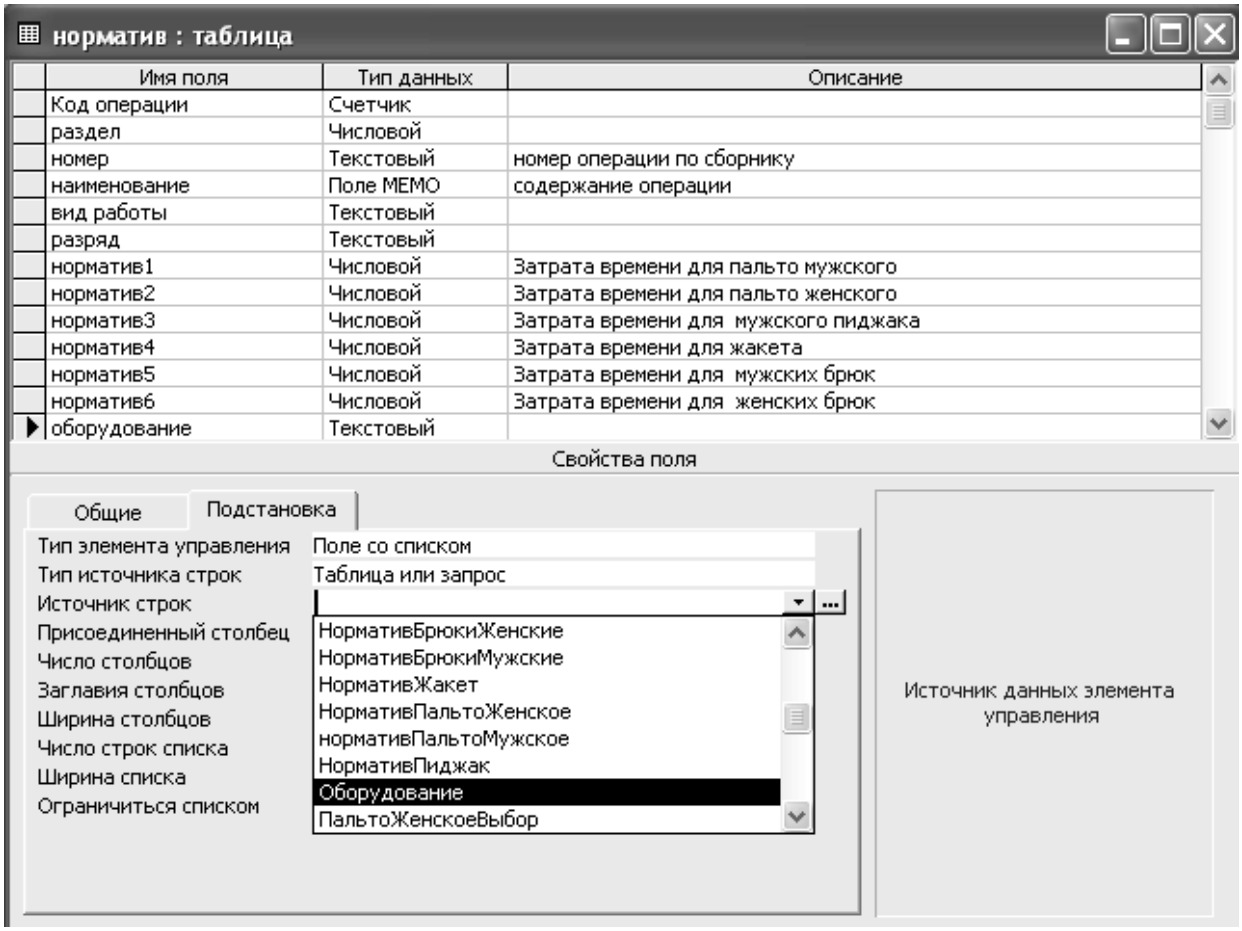


Рис. 7. Создание поля со списком

11. Сохраните структуру таблицы Норматив.
12. Перейдите в режим "**Вид**" и заполните таблицу Норматив новыми данными, выбирая оборудование из списка.

13. Проделайте действия по изменению данных для 25-30 строк таблицы **Норматив**. Закройте таблицу **Норматив**.
14. Добавьте в таблицу "**Разделы**" поле **Рисунок** тип, которого "**поле объекта OLE**". Сохраните таблицу.
15. Для ввода данных в таблицу "Разделы" создайте форму с аналогичным именем, для чего в разделе **Формы** нажмите кнопку **Создать** и выберите режим **Автоформа в столбец**, выбрав в качестве источника данных таблицу **Список**. Для возможной корректировки вида формы нажмите на кнопку **Конструктор**.
16. Покажите результат преподавателю.
17. Для сохранения формы нажмите на кнопку **Сохранить.**

ЗАДАНИЕ 2.Создание запросов.

1. Загрузите программу **ACCESS**. Откройте базу **Нормы времени**.
2. Создайте запрос на выборку с именем **Пальто Мужское** (рис.8).

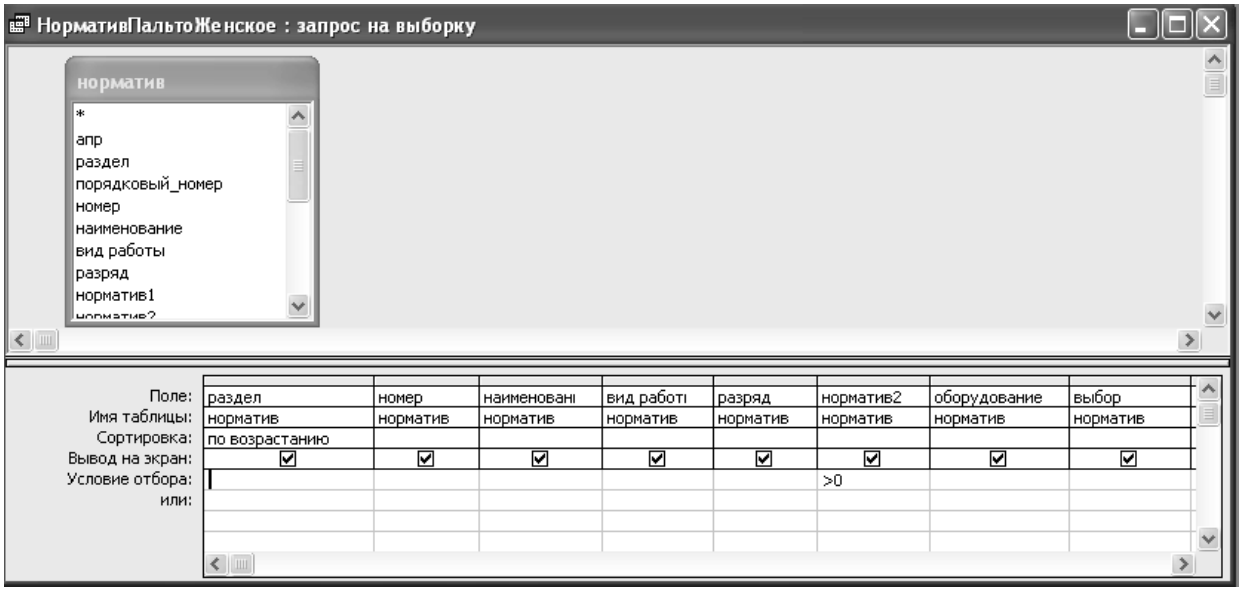


Рис.8. Конструктор простого запроса на выборку

3. Выберите закладку ***Запросы***, если находитесь в другом диалоговом окне.
4. Щелкните мышкой на кнопке ***Создание запроса в режиме конструктора***.

5. Добавьте таблицу ***Норматив***
6. Закройте окно ***Добавление таблицы***
7. Выберите поля ***Раздел, Номер операции, Наименование операции, Вид работ, Разряд, Оборудование, Выбор***.
8. Из четырех полей ***Норматив*** выберите только ***Норматив 1,*** соответствующий данным о пальто мужском**.** Для него установите ***условие >0.***
9. Для всех выбранных полей запроса установите галочку ***Вывод на экран.***
10. Щелкните по кнопке ! для представления запроса. Убедитесь, что из таблицы Норматив отобраны только нормы времени для мужского пальто.
11. Создайте запрос на выборку по разделам, имеющим отношение к мужскому пальто (рис.9).

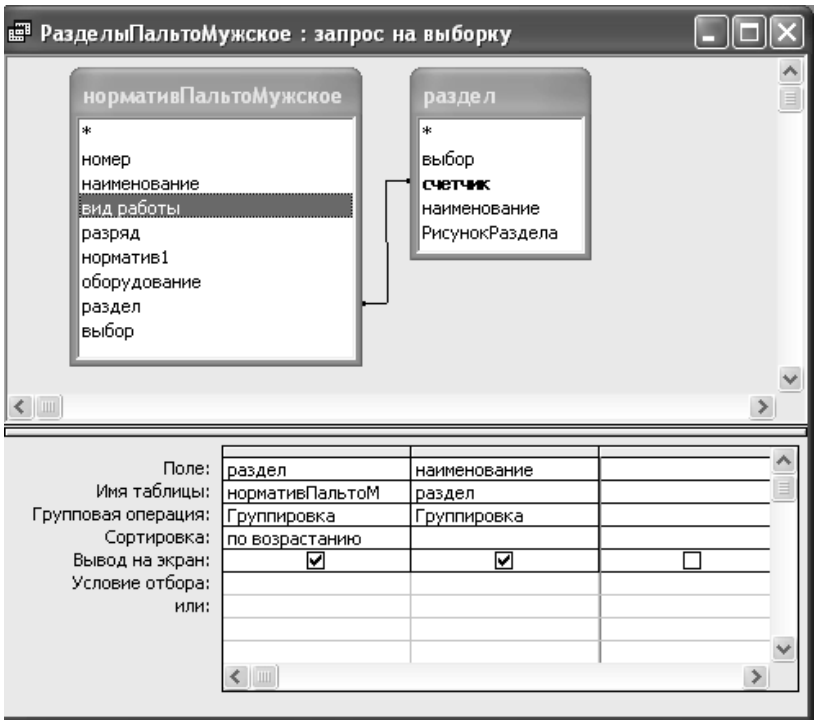


Рис.9. Конструктор запроса на выборку для групповых операций

12. Добавьте запрос ***НормативПальто Мужское*** и таблицу ***Разделы***
13. Закройте окно ***Добавление таблицы***
14. Из таблицы **Разделы** выберите поля ***Наименованиея, Раздел и Рисунок раздела***, а из запроса **НормативПальтоМужское -** ***Раздел***

15. В графе Групповая операция выберите **[Группировка]**
16. Отсортируйте поле «Раздел» а запросе Норматив Пальто Мужское по возрастанию
17. Сохраните запрос с именем Разделы Пальто Мужское
18. Просмотрите запрос

Создание запроса на выборку данных, удовлетворяющих условию (Рис.10).

Создадим запрос на создание таблицы для выбранных операций из запроса **Норматив Пальто Мужское**.

1. Добавьте запрос **Норматив Пальто Мужское**
2. Закройте окно ***Добавление таблицы/запроса***
3. Выберите поля ***Раздел, Номер операции, Наименование операции, Вид работ, Разряд, Оборудование, Выбор***.
4. Для поля Выбор установите ***условие: ДА.***
5. Для всех выбранных полей запроса установите галочку ***Вывод на экран.***
6. Щелкните по кнопке ! для представления запроса. Убедитесь, что в данный момент отобранных нет, т.е. таблица пуста.
7. Сохраните запрос под именем **Выбор Пальто Мужское**

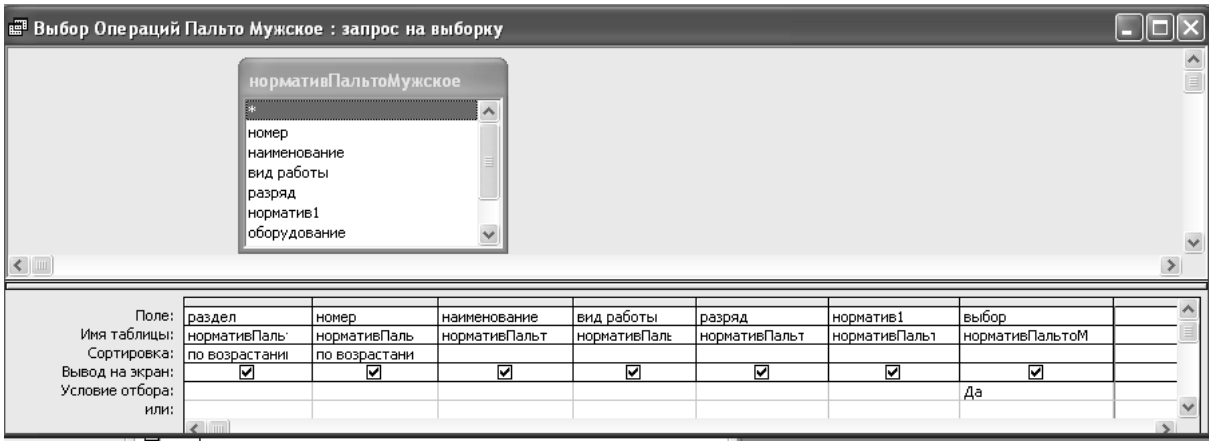


Рис.10. Запрос на выбранные операции (отбор по графе ***Выбор***)

8. Создание запроса на обновление данных (Рис.11).

Применяется для изменения большого объема данных удовлетворяющих определенным условиям, например, необходимо все значения ***ДА*** в логическом поле ***Выбор*** изменить на ***НЕТ***.

9. Добавьте таблицу ***Норматив***
10. Закройте окно ***Добавление таблицы***
11. Выберите только одно поле ***Выбор***.
12. Щелкните по стрелке рядом с кнопкой Тип запроса на панели инструментов и выполните команду **Запрос на обновление…**
13. Напечатайте имя таблицы **Новая Последовательность** и щелкните по кнопке ОК
14. Для обновления поля ***Выбор*** установите НЕТ**.**
15. Щелкните по кнопке ! для представления запроса.
16. Посмотрите результат и покажите преподавателю

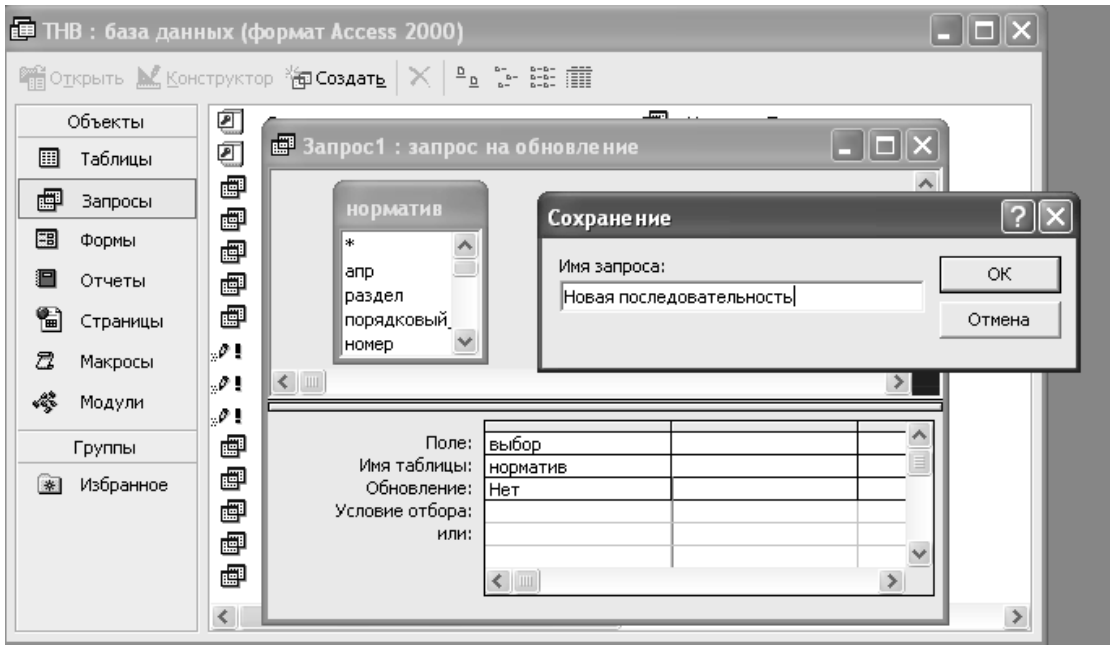


Рис.11. Построение запроса на обновление данных

ЗАДАНИЕ 3.Создание отчетов

1. Создайте с помощью **Конструктора** отчет **Технологическая последовательность**.

Ставится задача вывести выбранные технологические операции на печать. В качестве источника возьмем запрос **Выбор Пальто Мужское**.

1. Откройте закладку ***Отчеты*** и щелкните по кнопке ***Создать***.
2. В окне диалога ***Новый отчет*** выберите режим ***Конструктор*** и запрос **Выбор Пальто Мужское** в качестве источника данных. Щелкните по кнопке ***ОК***.
3. Если отсутствует раздел ***Заголовок отчета***, добавьте его с помощью команды Вид – Заголовок/примечание отчета.
4. Напечатайте в разделе заголовка отчета: **ТЕХНОЛОГИЧЕСКАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ НА ИЗГОТОВЛЕНИЕ МУЖСКОГО ПАЛЬТО**. Для этого щелкните по кнопке ***Аа – надпись*** на дополнительной **Панели элементов**. Укажите мышкой место для начала надписи и введите текст с клавиатуры. В конце ввода нажмите клавишу Enter. Надпись, приведенная в разделе заголовка отчета, будет единственной для всего отчета.
5. Сделайте надпись размером 16. Выполните команду ***Формат – Размер – по размеру данных***. Измените размер окна, если надпись видна не полностью.
6. Напечатайте в разделе ***Верхний колонтитул*** надпись ТЕХНОЛОГИЧЕСКАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ размером 14. Эта надпись будет появляться на каждой странице. Передвиньте надпись на правый край страницы.
7. Поместите поля *Номер операции, Содержание операции, Вид работ, Разряд, Норматив 1, Оборудование* в раздел **Область данных**. Оставьте подписи полей. Сделайте размер 12. Выполните команду ***Формат – Размер – по размеру данных***.
8. В разделе ***Нижний колонтитул*** поместите номера страниц. Для этого добавьте в область данных пустое поле, щелкнув по кнопке ***аб*** и перетащив ее мышкой в нужное место

9. Удалите подпись поля. Выделите пустое поле
10. Включите кнопку – ***Свойства***
11. Наберите значение =[***Page***] в строке ***Данные.***
12. Закройте текущее окно. Номера страниц будут появляться в конце каждой страницы (Рис.12).

Замечание. Выбрав режим предварительного просмотра документа, вы увидите его таким, каким он будет напечатан на бумаге. Линии сетки можно убрать (выберите пункт меню ***Вид*** *и отключите галочку у пункта меню* ***Сетка****).*

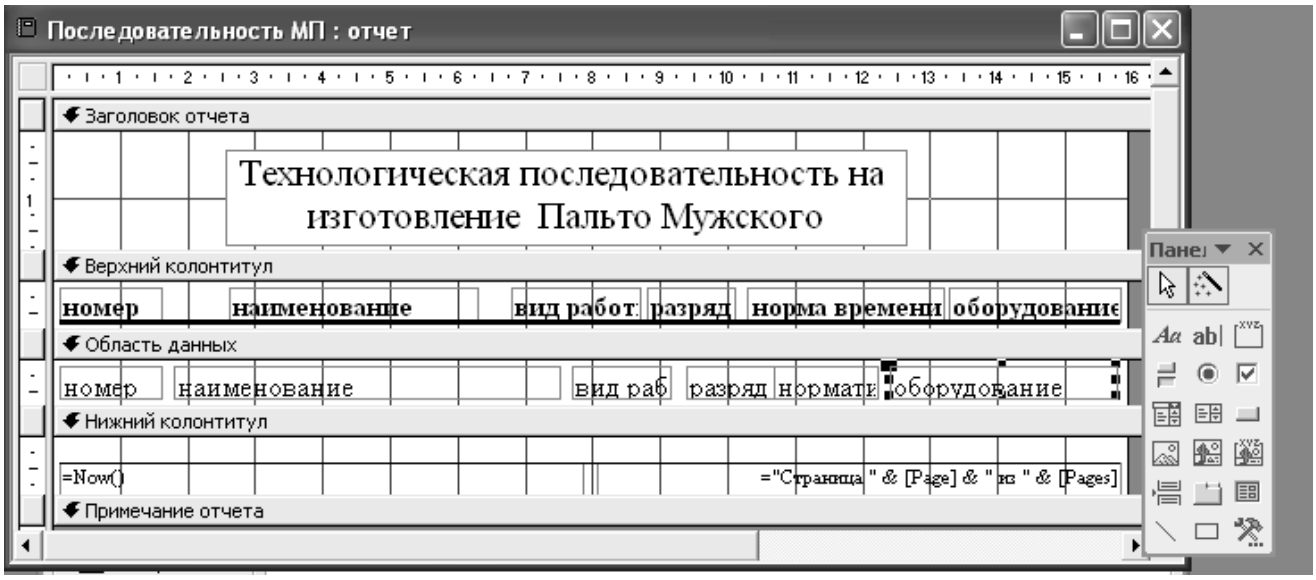
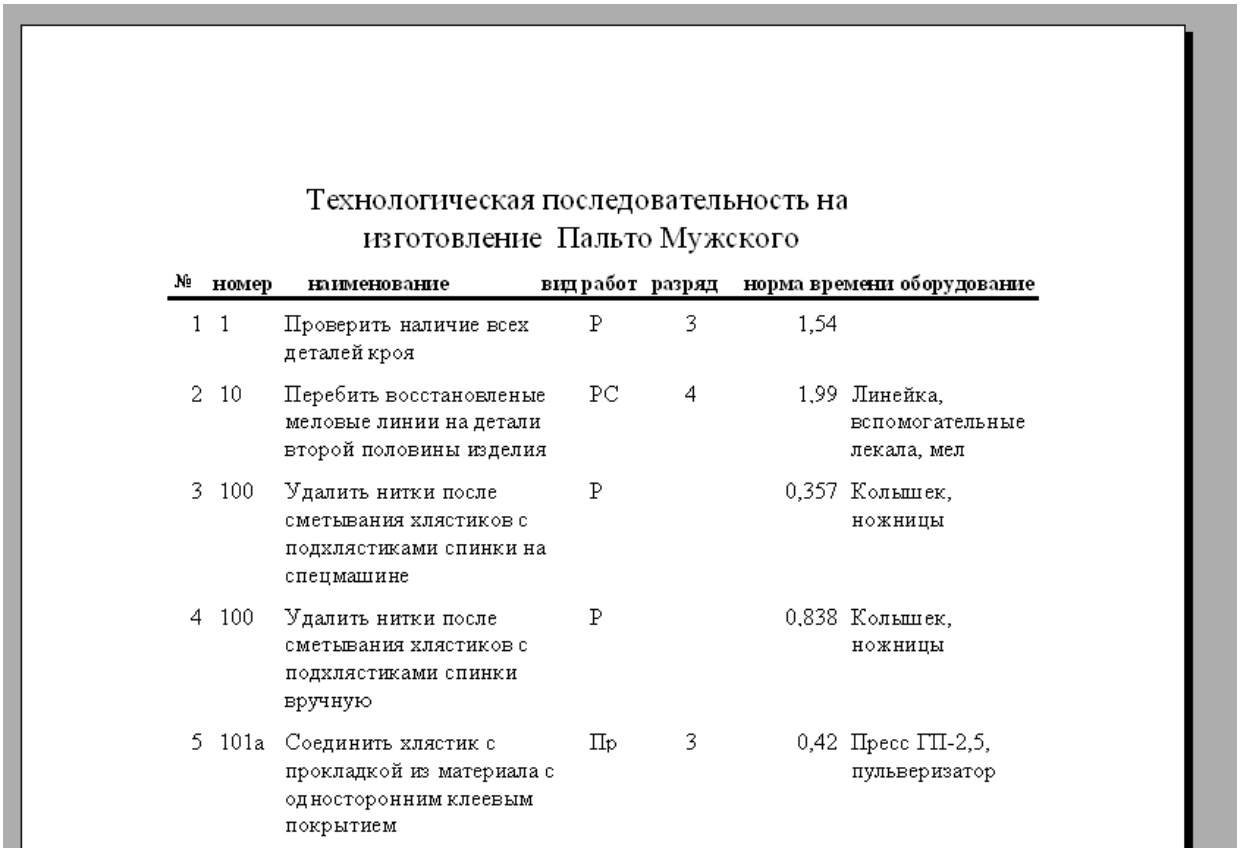


Рис. 12. Конструктор Отчета

2.Пронумеруйте записи в области данных отчета.

1. Установите для отчета режим Конструктор.
2. Добавьте в область данных пустое поле, щелкнув по кнопке ***аб*** и перетащив ее мышкой в область данных в положение перед полем ***Номер операции***.
3. Удалите надпись пустого поля.
4. Выделите поле и щелкните по кнопке ***Свойства*** на панели инструментов.
5. Введите в ячейку ***Данные*** выражение =1.

6. В ячейке свойства ***Суммы с накоплением*** выберите значение ***Для всего***. Во время предварительного просмотра или вывода отчета на печать номер на каждой следующей записи будет увеличиваться на 1, т.е. будет 1,2,3…
7. Закройте окно со свойствами полей
8. Перейдите в режим предварительного просмотра и просмотрите результат (рис.13). Определите, сколько операций вы отобрали, составляя последовательность.
9. Закройте отчет, предварительно сохранив его под именем ***Технологическая последовательность***.
10. Представьте отчеты преподавателю.



Технологическая последовательность на изготовление Пальто Мужского

№	номер	наименование	вид работ	разряд	норма времени	оборудование
1	1	Проверить наличие всех деталей кроя	Р	3	1,54	
2	10	Перебить восстановленые меловые линии на детали второй половины изделия	РС	4	1,99	Линейка, вспомогательные лекала, мел
3	100	Удалить нитки после сметывания хлястиков с подхлястиками спинки на спецмашине	Р		0,357	Колышек, ножницы
4	100	Удалить нитки после сметывания хлястиков с подхлястиками спинки вручную	Р		0,838	Колышек, ножницы
5	101а	Соединить хлястик с прокладкой из материала с односторонним клеевым покрытием	Пр	3	0,42	Пресс ГП-2,5, пульверизатор

Рис. 13. Просмотр Отчета

ЗАДАНИЕ 4.Создание форм. Создание макросов

- С помощью **Конструктора** создайте форму **Норматив Пальто Мужское**

Ставится задача вывести массивы с технологическими операциями по каждому изделию в отдельности. В качестве источника возьмем запрос **Норматив Пальто Мужское**

17. Откройте закладку ***Формы*** и щелкните по кнопке ***Создать***
18. В окне диалога ***Новая форма*** выберите режим ***Ленточная*** и запрос **Норматив Пальто Мужское** в качестве источника данных. Щелкните по кнопке ***ОК (Рис.14).***

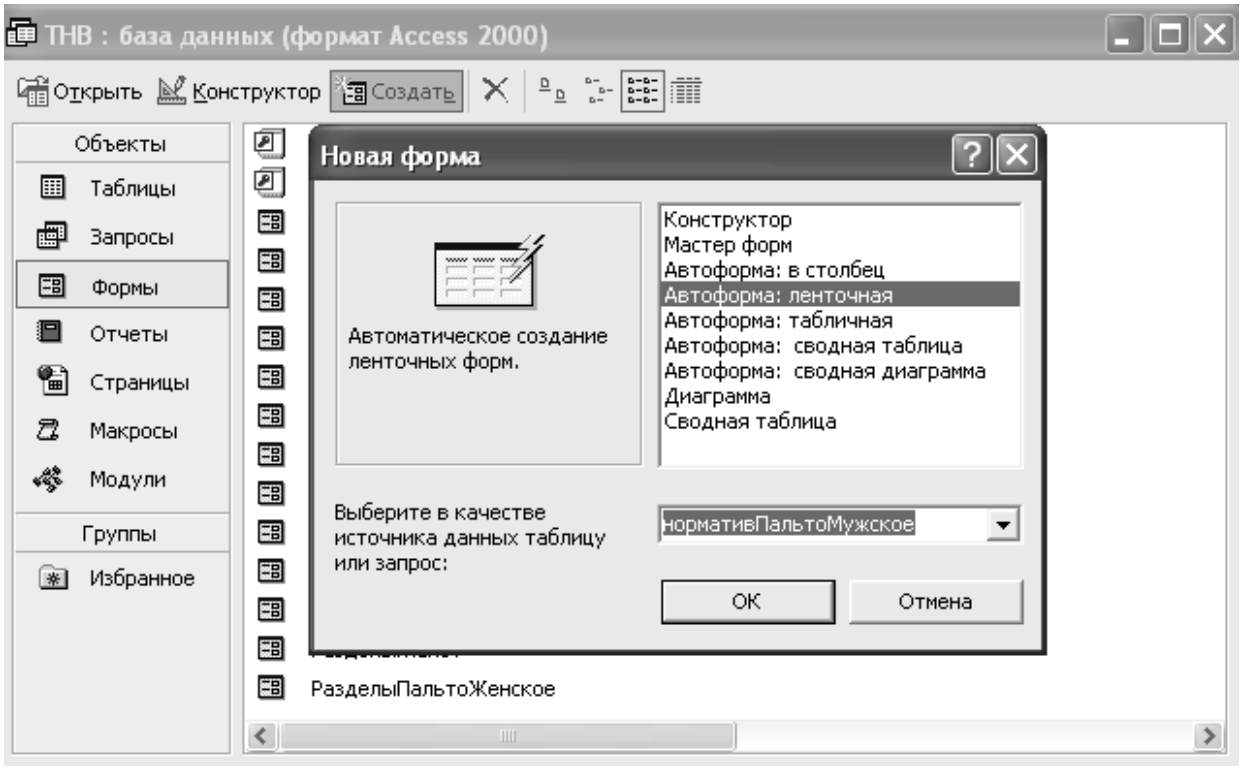


Рис. 14. Выбор автоматического создания формы

Полученная в результате автоматического создания форма выглядит крайне некрасиво (Рис.15). По сути своей являясь отображением данных запроса **Норматив Пальто Мужское,** эта форма представляет все графы таблицы так, в каком порядке они были выбраны в конструкторе этого запроса. Поэтому рекомендуется либо создать форму заново, на вручную, в режиме конструктора или «подогнать» и пересортировать все ячейки полученной автоформы.

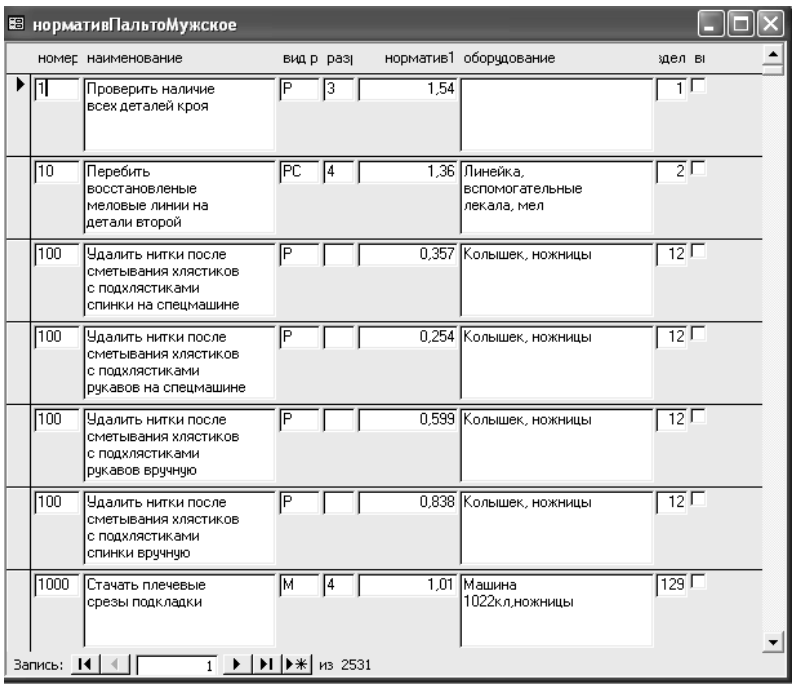


Рис.15. Внешний вид автоформы ленточного типа

19. Перейти в режим Конструктора формы (рис.16). Убедиться в наличии ***Панели инструментов Форм***, при необходимости включить через меню **Вид**.

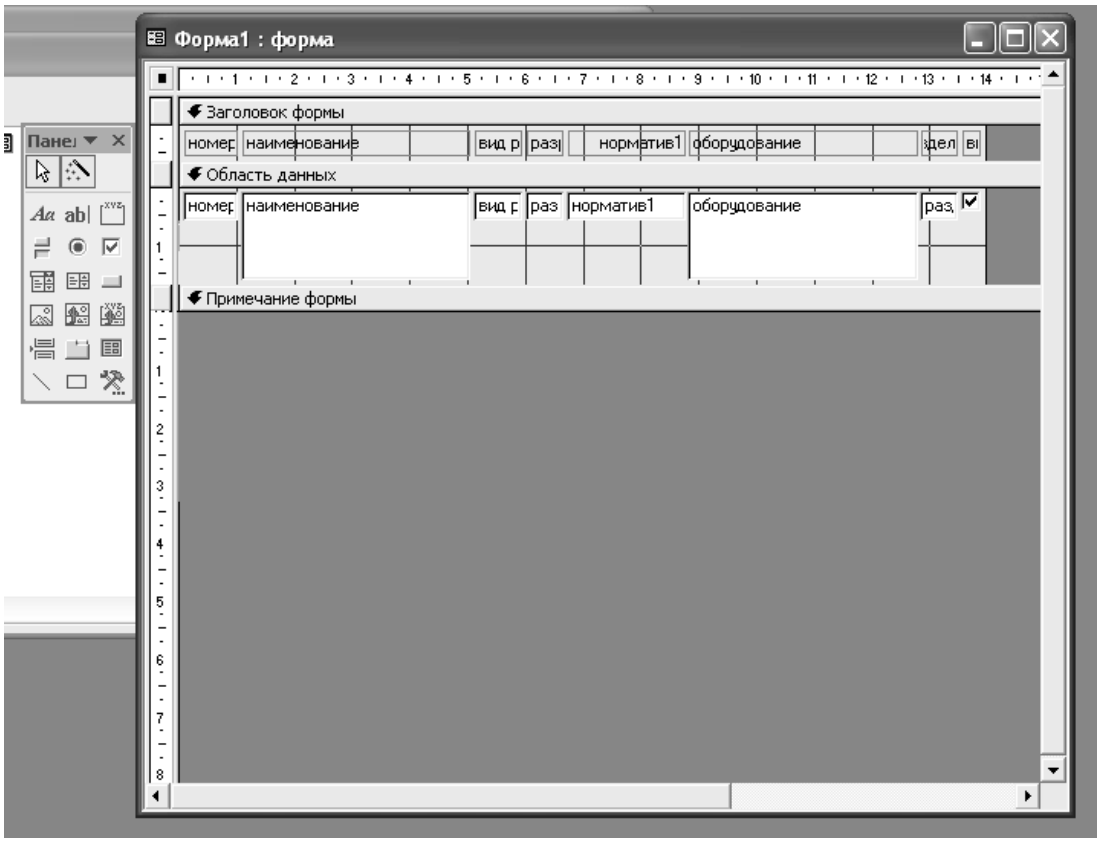


Рис. 16. Конструктор форм

20. Сделайте область формы шире, почти на весь экран, для чего измените размер окна, потянув правую границу мышкой (Рис.17).

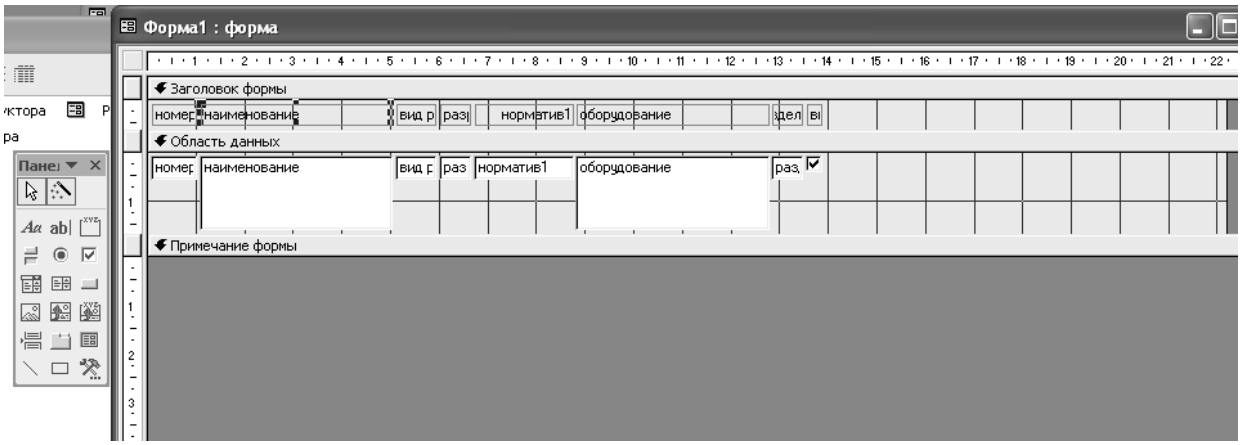


Рис. 17. Изменение ширины формы

21. Перетащите ячейки формы и измените их размеры и местоположение с помощью мышки так, чтобы они занимали всю ширину окна. Заголовком такой формы является шапка таблицы, поэтому не следует забывать при перетаскивании граф с данными (Область данных) заодно перемещать и названия граф.
22. Измените размер шрифта на 12 заголовка формы (Наименование граф таблицы). Добавьте область Примечания Формы, потянув мышкой вниз (Рис.18).

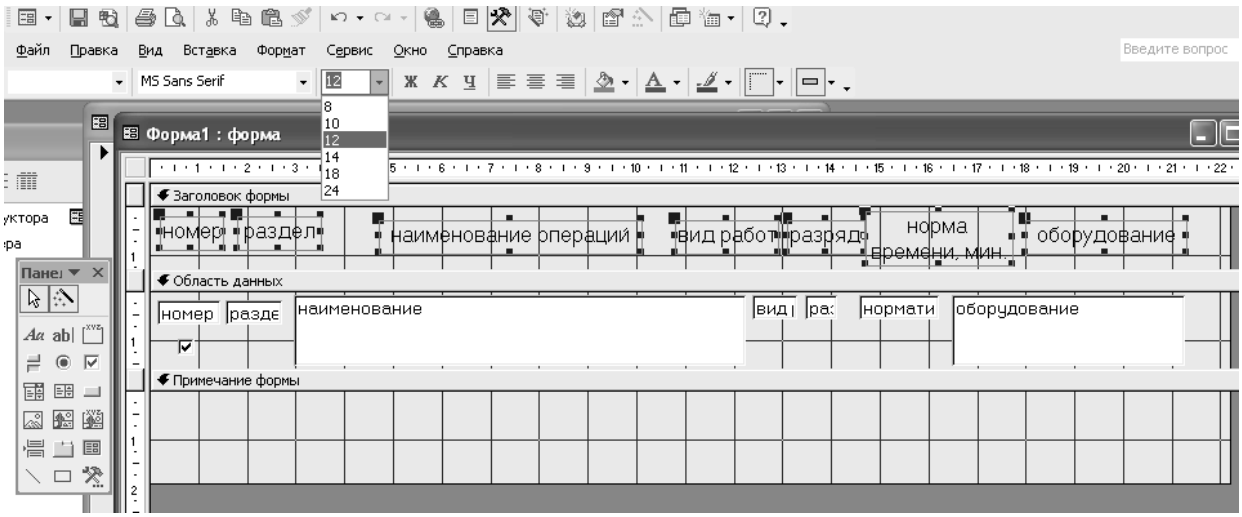


Рис. 18. Конструктор формы в новом ее виде

23. Просмотрите, как теперь выглядит форма (Рис.18). Убедитесь, что при просмотре не «обрезается» ни одна из строк ленточной формы, как это имеет место быть на рис.19. Для избежания этого необходимо в конструкторе формы немного изменить ширину **области Примечания Формы**.

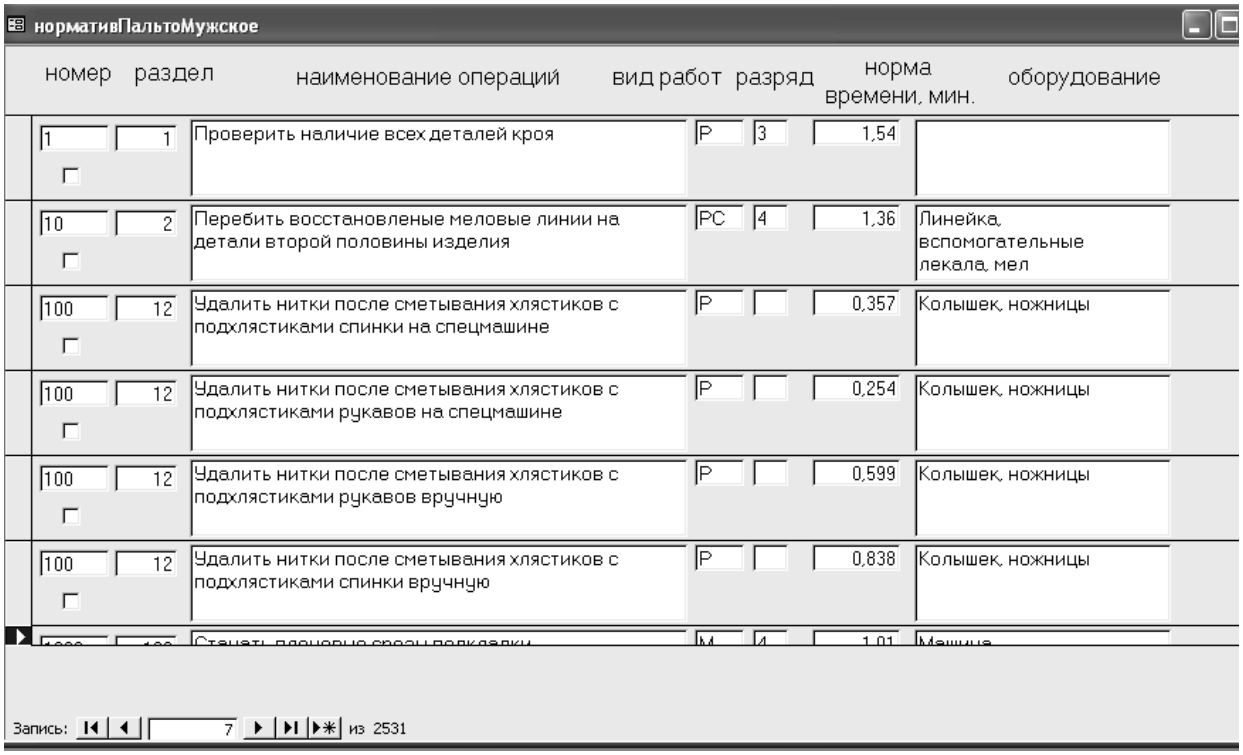


Рис. 19. Просмотр формы

24. Сделать 4кнопки на **области Примечания Формы**. **Первая** – Очистить данные в графе Выбор (Новая последовательность), **вторая** – запуск запроса Выбор Операций Пальто Мужское, **третья** – запуск формы Разделы Пальто Мужское, **четвертая** – закрытие этой формы.
25. Кнопки создаются путем запуска мастера, где в диалоговом режиме выбираются действия, которые следуют при нажатии кнопки (См. Приложение 6 и рис. 20). В результате форма должна выглядеть так, как показано на рис. 21.

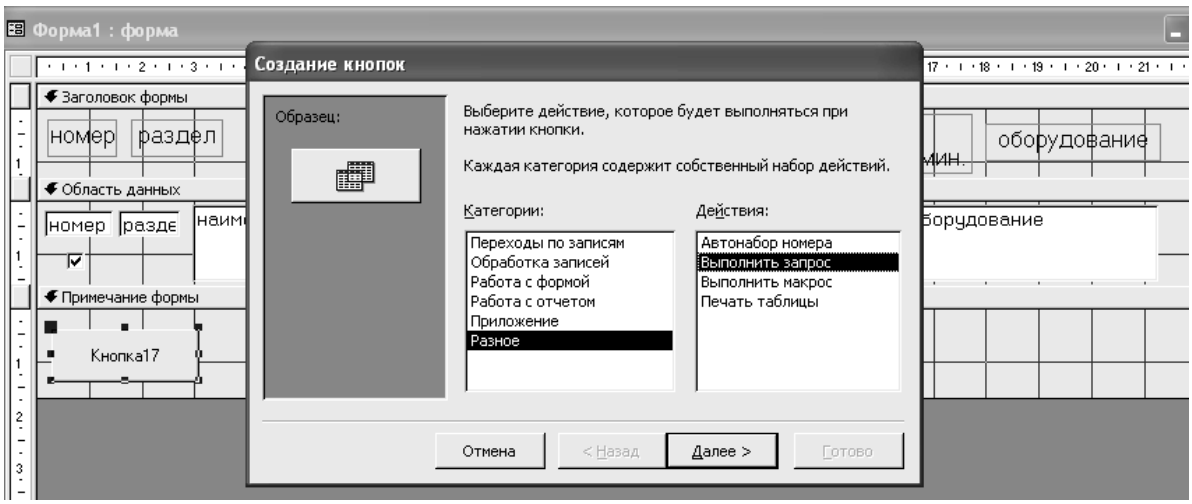
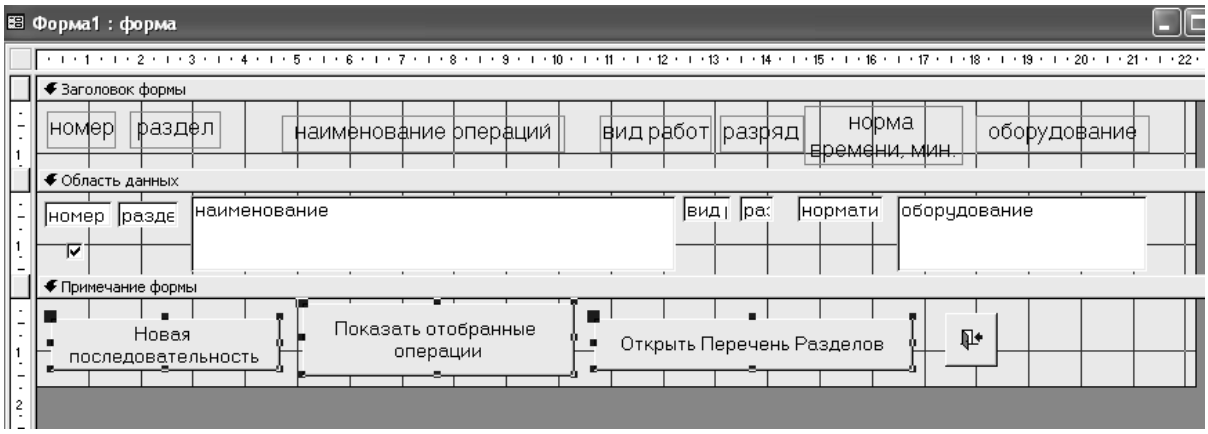
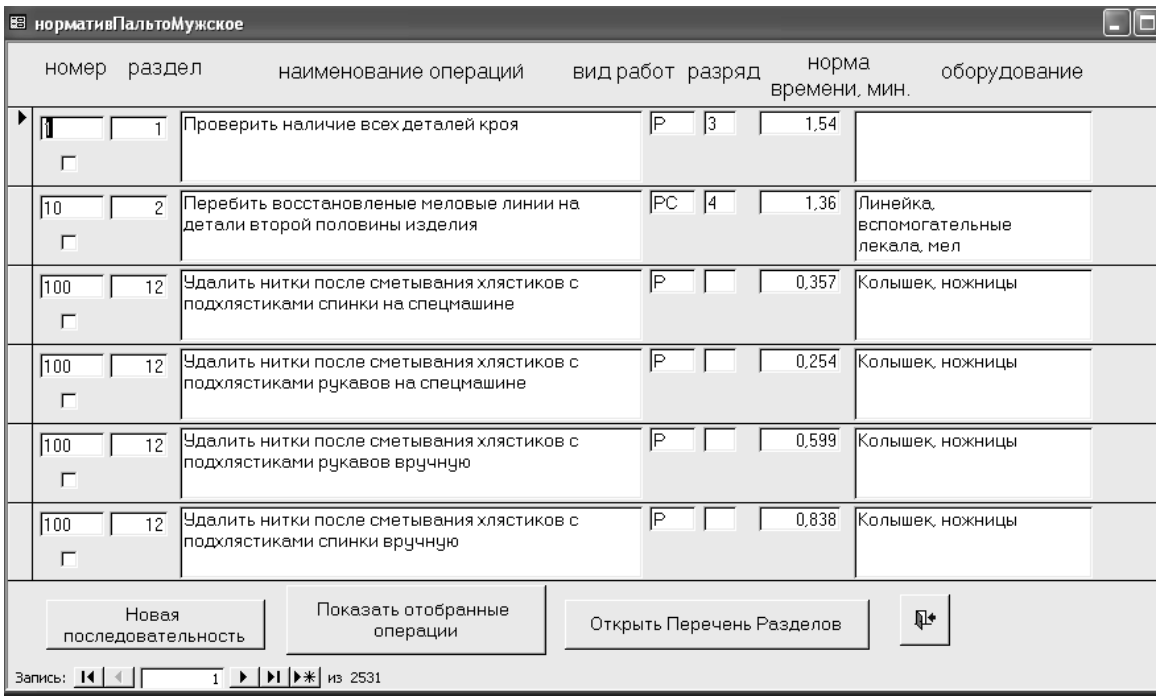


Рис. 20. Мастер создания кнопок

26. Протестируйте полученную форму. Для этого очистите графу Выбор (нажать первую кнопку), вызовите перечень разделов, для раздела с номером 39 выберите несколько операций, проставляя галочки в графе Выбор. Просмотрите отобранные Вами операции (кнопка 2).
27. Представьте работу преподавателю
28. Разработайте аналогичные запросы и формы для еще трех видов изделий основной таблицы Норматив. В результате получите всего четыре формы; три последних назовите: **Норматив Пальто Женское**, **Норматив Пиджак, Норматив Жакет.**



а) Конструктор готовой формы



б) Готовая к работе Форма

Рис. 21. Готовая форма в режиме конструктора и просмотра

- Создайте макросы, вызывающие формы: всего 4 макроса, каждый из которых вызывает определенную форму.

Ставится задача разработать микрокоманды для управления кнопками на главной форме, которую мы создадим в заключении работы. В качестве примера возьмем форму **Норматив Пальто Мужское**

29. Откройте закладку ***Макросы*** и щелкните по кнопке ***Создать***
30. В окне диалога ***Создать макрос*** выберите макрокоманду из раскрывающегося списка(рис.22); в качестве Аргумента макрокоманды выберите Форму Норматив Пальто Мужское. Сохраните макрос, дав ему имя аналогичное имени формы**.**

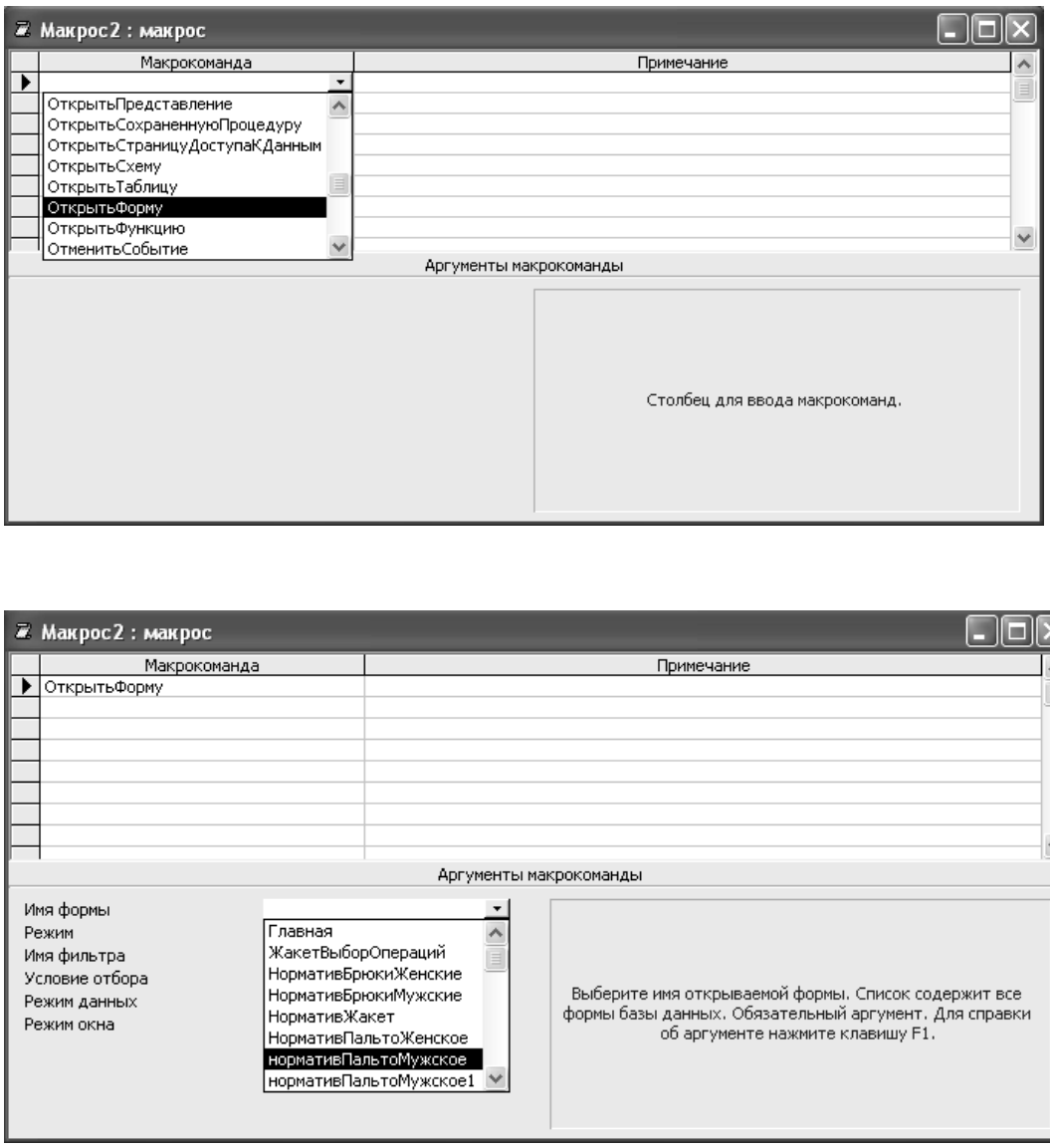


Рис. 22. Создание макроса «Открыть форму»

Самостоятельная работа

1. Аналогично создайте макросы **Норматив Пальто Женское**, **Норматив Пиджак, Норматив Жакет.**

2. Разработайте главную форму для базы данных.

•В режиме Конструктор форм вычертите произвольных размеров прямоугольник, сделайте на нем заголовок, соответствующий теме работы, в Примечании формы укажите фамилию, курс, группу**.**

•Создайте четыре кнопки, открывающие созданные вами формы. Запуск кнопки проведите с заданием макрокоманды (запуском макроса).

•Правой кнопкой мыши откройте меню и выберите **Свойства Формы**. В закладке Макет уберите ***Полосы прокрутки***, ***Область выделения*** и ***Кнопки перехода***.(Рис. 23)

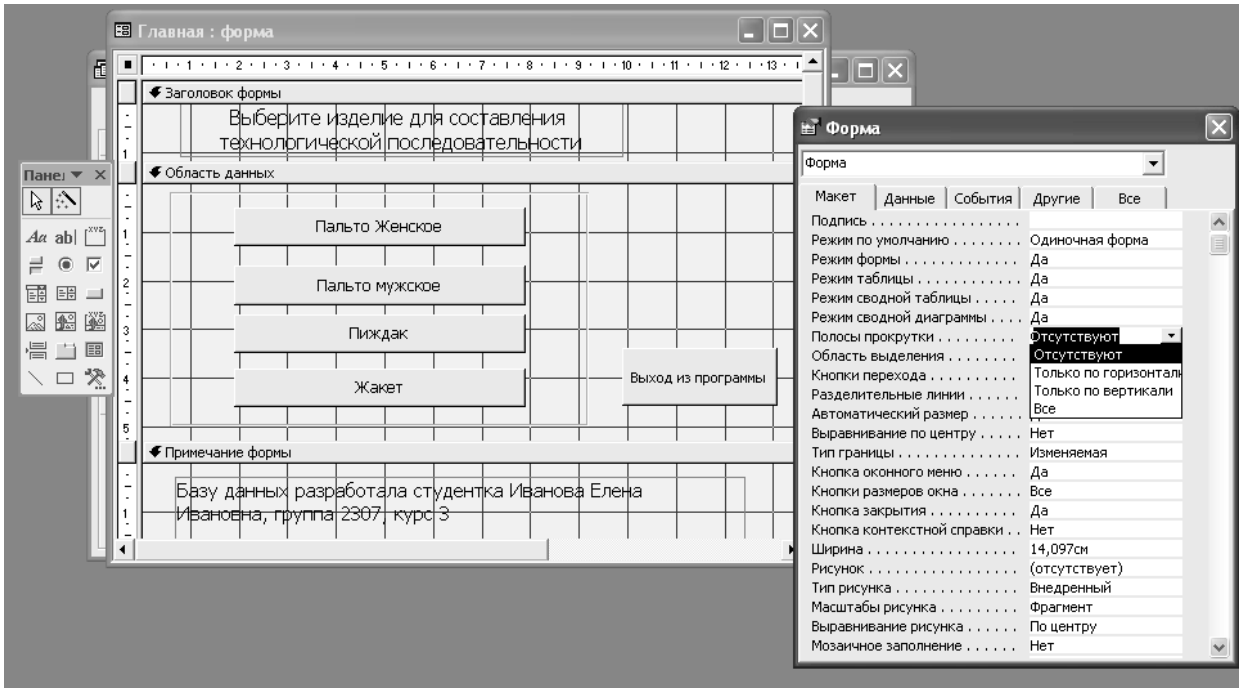


Рис. 23. Конструктор Главной формы базы данных

•Создайте кнопку для закрытия (Выход из программы) главной формы, всей базы данных и приложения **ACCESS** (используйте мастер кнопок или заранее сделанный макрос).

•Сохраните главную кнопочную форму, дав ей название **Главная**.(Рис.23)

•Для запуска приложения и одновременного открытия главной кнопочной формы создайте макрос ***Autoexec***. Макрос именно с таким именем открывает базу данных, при этом он должен и выполнять команду по открытии формы Главная.

•Представьте главную кнопочную форму преподавателю.

ЛАБОРАТОРНАЯ РАБОТА №19

НАВИГАЦИЯ И ПОИСК ИНФОРМАЦИИ В ИНТЕРНЕТЕ

Цель работы: получить практические навыки по навигации и поиску информации в Интернет, уметь подключать соединение для связи с Интернетом , настраивать программу-браузер.

Должны знать: основные информационные и информационно-поисковые русскоязычные ресурсы в Интернет, способы и приемы поиска информации в них, способы защиты и обеспечения конфиденциальности информации при работе в Интернет.

Должны уметь: грамотно работать с программой-браузером выполнять основные ее установки, создавать закладки в Избранном, создавать брандмауэр (защиту для компьютера, работающего в Интернет).

Методические указания.

Всемирная паутина — это вольный перевод английского словосочетания *World Wide Web,* которое часто обозначается как WWW или Web. Бурное развитие сети Интернет, которое происходило на протяжении 90-х годов, в первую очередь обусловлено появлением новой технологии WWW.

Технология WWW (ПРИЛОЖЕНИЕ 10) позволяет создавать ссылки (их также называют гиперссылками), которые реализуют переходы не только внутри исходного документа, но и на любой другой документ, находящийся на данном компьютере и, что самое главное, на любой документ любого компьютера, подключенного в данный момент к Интернету.

Для просмотра Web-страниц используют специальные программы – браузеры. В настоящее время наиболее распространенными браузерами являются Internet Explorer (его русскоязычная версия часто называется Обозреватель) и Netscape Communicator (Коммуникатор).

Задание №1. Настройка браузера Internet Explorer

4.Запустить Internet Explorer После запуска откроется окно Обозревателя, в которое будет загружена начальная Web-страница. По умолчанию в русскоязычной версии Internet Explorer такой страницей является страница Web-сервера фирмы Microsoft.

5.Окно браузера (рис.24) содержит меню, панель инструментов, окно *Адрес,* а также рабочую область, в которой и просматриваются Web-страницы. Окно *Адрес* позволяет ввести с клавиатуры или выбрать из списка URL нужной Web-страницы. Панель инструментов позволяет переходить с одной Web-страницы на другую (кнопки *Вперед, Назад, Домой),* управлять процессом загрузки (кнопки *Остановить, Обновить)* и др.

6.Внешний вид и местоположение панелей инструментов можно изменять, перетаскивая компоненты панели инструментов с помощью мыши или используя меню *Вид*. Параметры просмотра Web-страниц в браузере можно изменять с помощью многочисленных настроек меню Сервис. Так, можно изменить адрес начальной страницы, загружаемой в Обозреватель.

7.В меню *Сервис* выбрать пункт *Свойства обозревателя* (Рис.25). Откроется диалоговая панель *Свойства обозревателя.* На вкладке *Общие* в группе *Домашняя страница* в поле *Адрес:* ввести URL нужной страницы, (Рис. 25). Например, адрес начальной страницы Web-сайта, представляющего один из самых мощных поисковых, развлекательных, коммерческих и информационных серверов Рунета – Яндекс: http://yandex.ru. Чтобы убедиться, что введен правильный адрес, щелкнуть по кнопке *Домой.* Обозреватель должен загрузить начальную страницу информационного ресурса Яндекс.

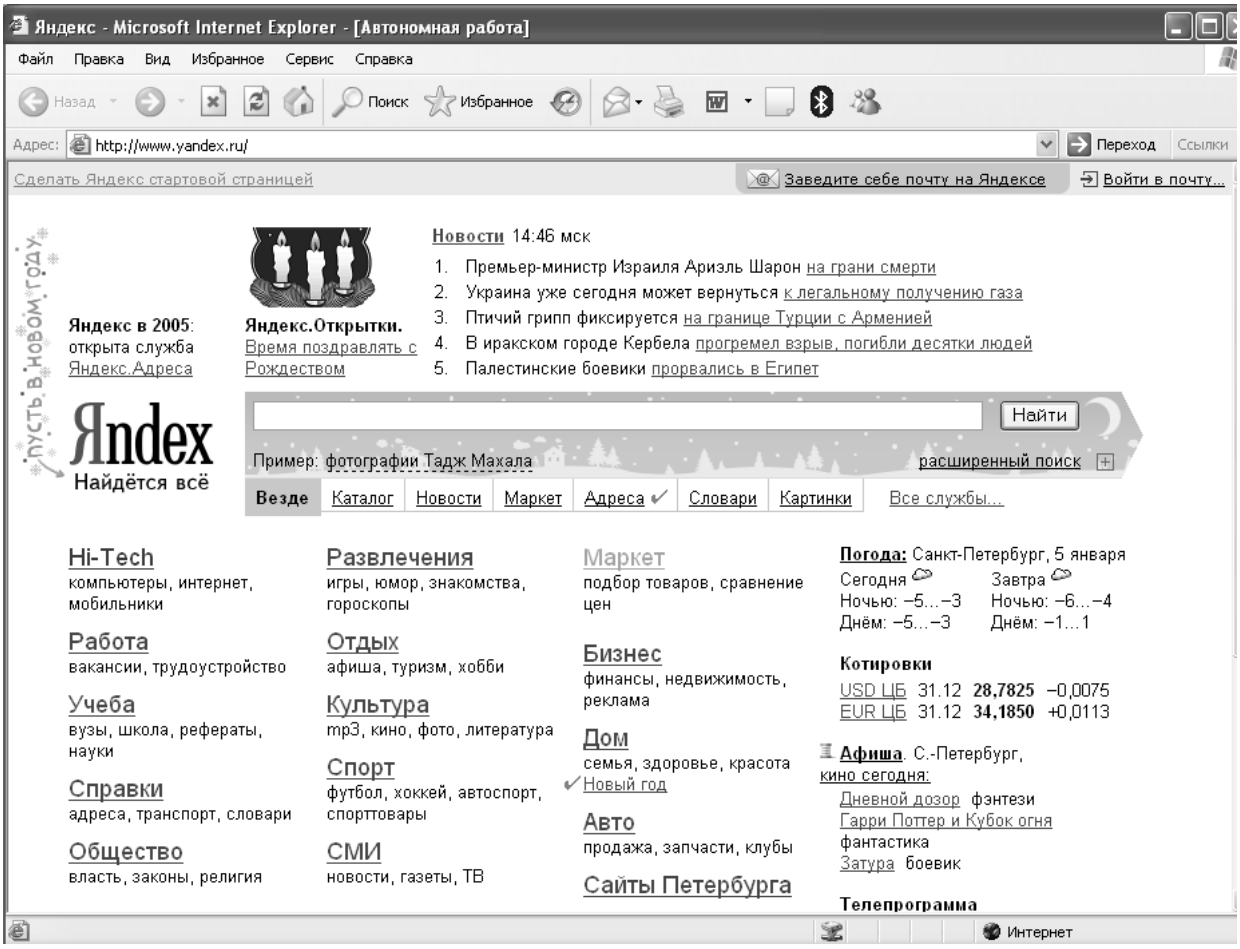


Рис.24. Окно браузера Internet Explorer

8.Большое значение имеет настройка Обозревателя на просмотр Web-страницы в правильной кодировке, то есть той кодировке, в которой Web-страница была создана. В большинстве случаев Обозреватель автоматически определяет кодировку и, соответственно, правильно отображает Web-страницу. Однако в некоторых случаях пользователю необходимо настроить Обозреватель на требуемую кодировку вручную. Например, для просмотра Web-страницы в кодировке *Windows* необходимо ввести команду [Вид-Вид кодировки-Кириллица (Windows)]. (Рис.26).

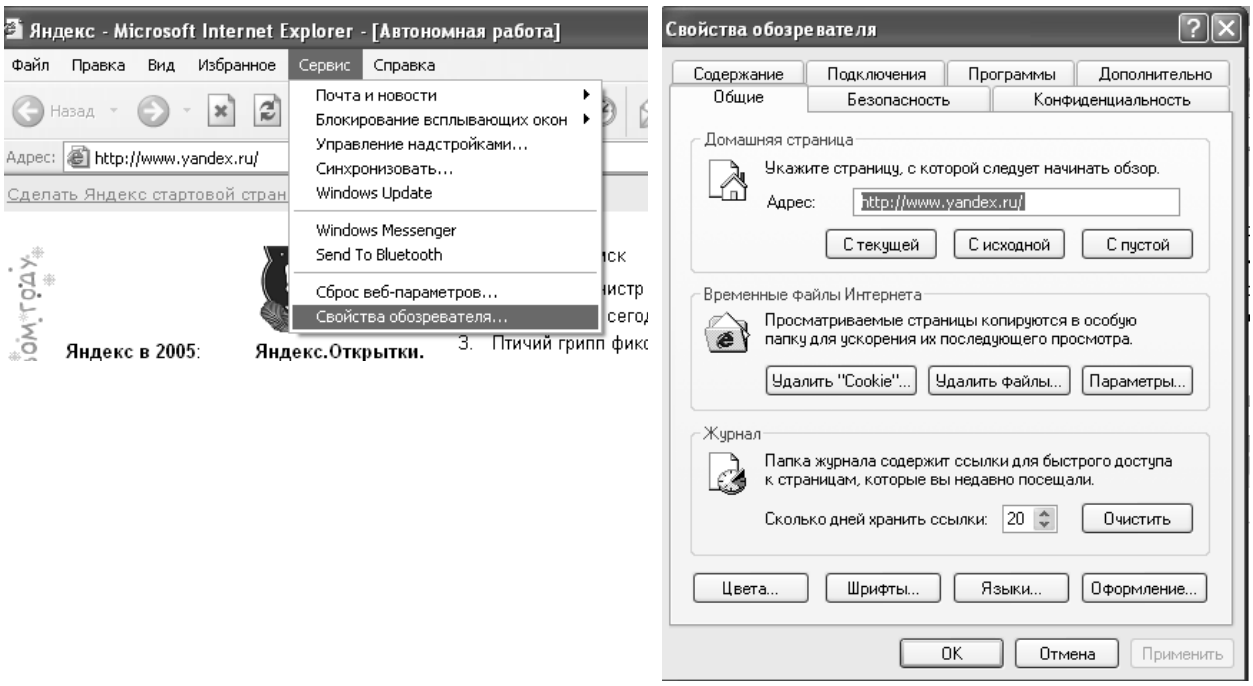


Рис.25.Вызов настройки свойств обозревателя. Изменение адреса «домашней» страницы

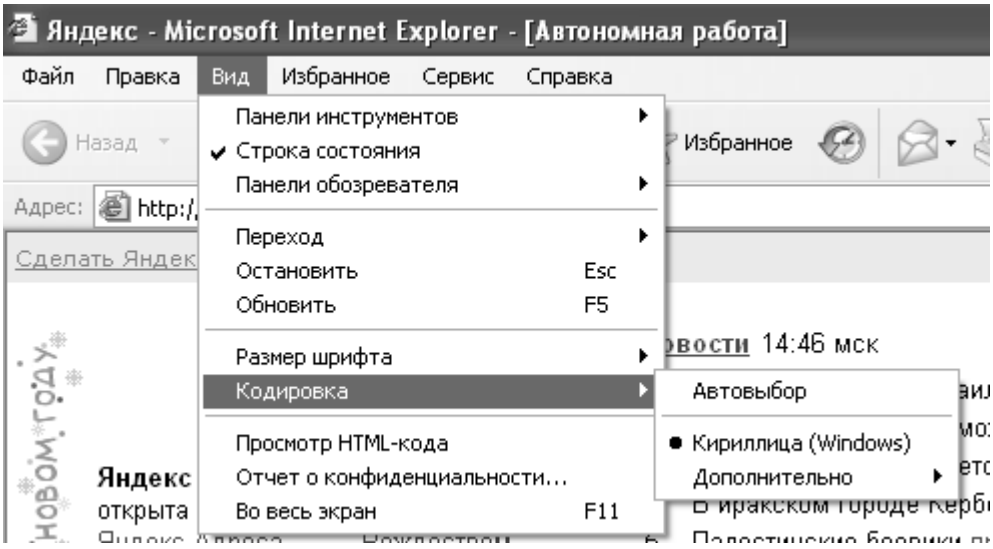


Рис. 26. Выбор кодировки для отображения страниц Интернета

9.Можно ускорить процесс загрузки Web-страниц в случае соединения с Интернетом на низкой скорости передачи информации (16,4 Кбит/с и менее) или в случае перегруженности Web-страниц мультимедийными объектами, имеющими большой информационный объем. Для этого необходимо отключить загрузку мультимедиа объектов

(рисунки, анимация, звук, видео). Ввести команду [Сервис-Свойства обозревателя...], на появившейся диалоговой панели *Свойства обозревателя* выбрать вкладку *Дополнительно.* С помощью прокрутки найти в окне раздел *Мультимедиа* и снять все флажки этого раздела (Рис.27)

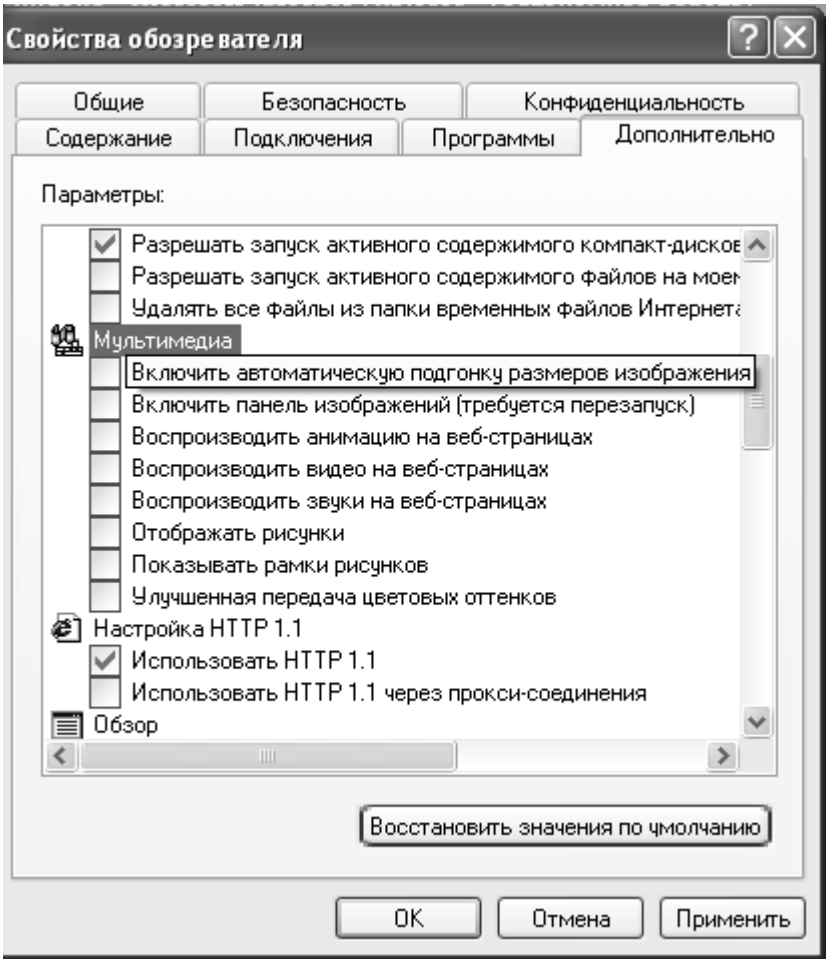


Рис. 27.Отключение загрузки мультимедийных объектов

Задание №2. Путешествия по Всемирной паутине. Поиск и информации и навигация.

Для того чтобы начать путешествие по Всемирной паутине, необходимо подключиться к Интернету и запустить какой-нибудь браузер, например, Internet Explorer. После загрузки начальной (домашней) страницы можно поступать различными способами:

1. воспользоваться ссылками загруженной Web-страницы брау-

зера;

2. в строку *Адрес* ввести адрес (URL) интересующей Web-страницы;
3. воспользоваться «закладками» Web-страниц

1. ***Использование ссылок.*** Путешествие по Всемирной паутине можно начать с путешествия по страницам сайта. Web-сайты обычно содержат достаточно большое количество страниц, поэтому для переходов с одной страницы на другую внутри сайта используется *панель навигации.* Панель навигации содержит названия основных разделов сайта.

Загрузить начальную страницу Web-сайта по заданию преподавателя. Загрузится страница, содержащая *панель навигации,* которая расположена в левой части страницы.

2. ***Ввод URL в окне*** *Адрес.* Пусть нас интересует информация о способах конструирования одежды. Достаточно легко «вычислить» имя русскоязычного Web-сервера, на котором можно найти такую информацию. Скорей всего должен быть Web-сервер, который относится к домену верхнего уровня ru. Фирма зарегистрировала имя домена второго уровня, скорее всего, совпадающее с ее названием, получаем legprombiznes.ru. Наконец, большинство Web-серверов имеют имя www, получаем полное имя сервера

www.legprombiznes.ru

Если набрать в поле *Адрес* URL данного сервера: http://www.legprombiznes.ru, то произойдет загрузка интересующей нас Web-страницы.

В Приложении 11 представлен небольшой список адресов, по которым можно найти в Интернет интересную информацию о современных системах проектирования одежды.

3. ***Создание и использование «закладок»***. В процессе чтения книги (учебника, справочника, энциклопедии) достаточно часто требуется вернуть-

ся к прочитанному материалу. Для более быстрого поиска нужной страницы часто в книгу вставляют закладку. В процессе путешествий по Всемирной паутине также целесообразно оставлять «закладки» на интересных Web-страницах, так как найти нужную страницу среди десятков миллионов Web-страниц гораздо труднее, чем найти страницу в книге.

Для создания закладки ввести команду [Избранное-Добавить в избранное...], на появившейся диалоговой панели *Добавление в избранное* выбрать название закладки (по умолчанию оно совпадает с названием Web-страницы) и щелкнуть по кнопке *ОК* (Рис.28). Теперь для загрузки данной страницы достаточно в пункте меню *Избранное* выбрать название закладки.

Самостоятельная работа

Согласно заданию преподавателя посетите не менее трех-четырех сайтов из Приложения 11 для поиска необходимой информации. Наиболее интересные страницы поместите в избранное, предварительно создав там соответствующую папку. Перенесите 3-4 страницы для просмотра в MS WORD.

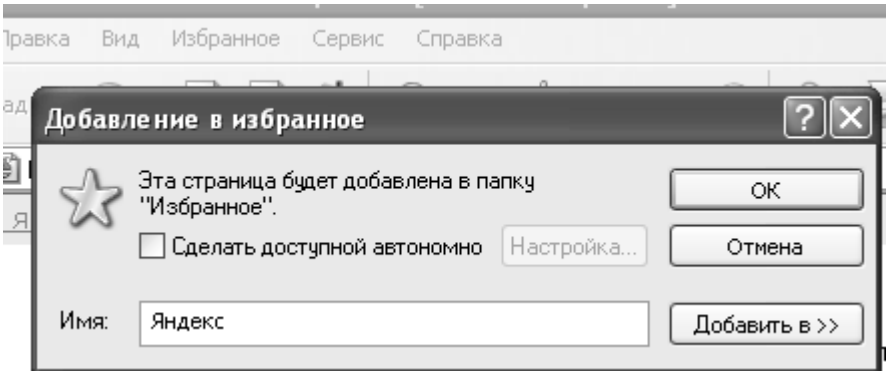


Рис.28. Добавление новой закладки в «Избранное»

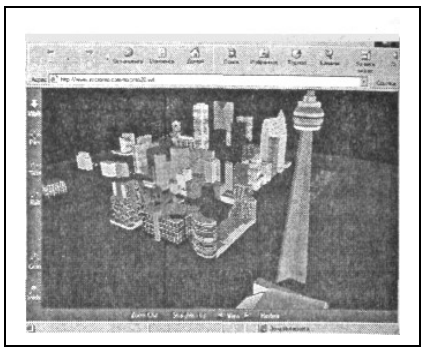
4. **Путешествия по виртуальной реальности.** С помощью специального языка моделирования виртуальной реальности (Virtual Reality Modeling Language — *VRML)* можно создавать виртуальные трехмерные миры, в которых можно затем перемещаться в различных направлениях и рассматривать

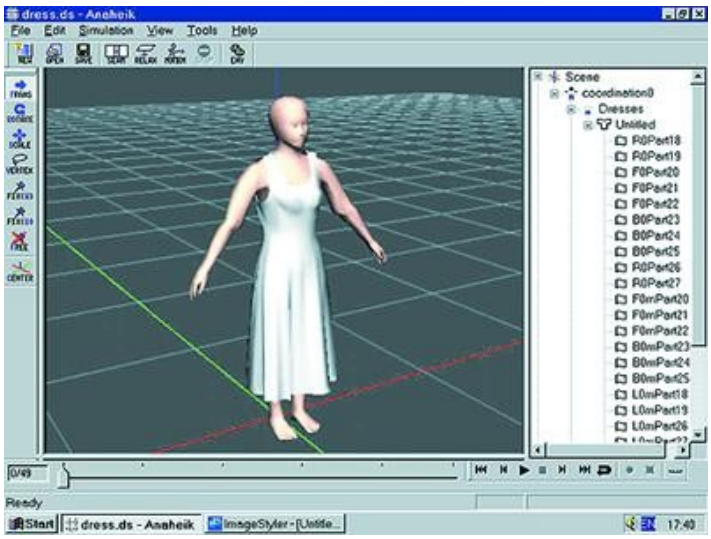
предметы с различных сторон. В Интернете существует достаточно много серверов, содержащих виртуальные миры, и в частности виртуальные города мира, по которым можно совершать виртуальные экскурсии (например, сервер www.intoronto.com — рис.29).

Еще один интересный виртуальный мир представлен японской фирмой Toyobo в программном пакете LookStailor. Программа одевает изделие не в статике на неподвижный манекен, а на движущуюся фигуру, то есть формирует своеобразный фильм, где «виртуальная модель» дефилирует в изделии по «виртуальному подиуму», причем очень реалистично. Здесь можно увидеть настоящий Дом Мод: манекенщицу можно переодеть в различные одежды, и она будет их демонстрировать, расхаживая по виртуальному подиуму. Такие виртуально-познавательные развлечения можно увидеть на http://www.dressingsim.com (рис.29).

Немного подробнее о последнем ресурсе.

В программе в основном используют 3-х мерные манекены. Хотя в принципе одеть можно все что угодно, будь то манекен человека, собаки или простой геометрический объект. Кроме того, одевать можно несколько манекенов сразу и в несколько пакетов одежды. В-третьих, можно задавать свойства ткани отдельно каждому лекалу. И, наконец, можно задать параметры внешней среды (ветер), после чего программа сформирует указанное количество кадров, и Вы увидите, как изделие развевается на ветру. И это еще не все. Есть возможность увидеть, где и насколько изделие прилегает к телу, как и в каком месте растягивается и изгибается ткань - причем в любой позиции при движении манекена. То есть возможности более чем впечатляющие.





Виртуальное путешествие по городу

Виртуальное переодевание манекена

Рис. 29. Виртуальная реальность во Всемирной Паутине

Для посещения виртуальных миров необходим специальный программный модуль, который подключается к браузеру и позволяет просматривать анимированную графику сцен виртуальной реальности. Наиболее распространенным является CosmoPlayer. Следует заметить, что для переодевания манекена и просмотра виртуальной демонстрации мод такой программный продукт не требуется.

ЛАБОРАТОРНАЯ РАБОТА №20

РАБОТА С ЭЛЕКТРОННОЙ ПОЧТОЙ И В ТЕЛЕКОНФЕРЕНЦИЯХ

Цель работы: получить практические навыки по работе с электронной почтой и в телеконференциях. Научиться работать с почтовыми программами, уметь их настраивать.

Должны знать: правила пользования электронной почтой, этику общения в телеконференциях по темам; как создать почтовый электронный ящик на бесплатном российском сервере, как установить почтовую программу на примере MS Outlook.

Должны уметь: правильно работать с программой Outlook, выполнять основные установки, в том числе коммутацию с Интернет, создавать электронный почтовый ящик, настраивать его.

Методические указания.

За последнее время для работы с электронной почтой стали часто использоваться web-технологии. Появились web-сайты, которые предлагают всем желающим зарегистрировать свой бесплатный электронный почтовый ящик. Примером таких сайтов могут являться http://mail.ru, http://www.rambler.ru, http://www.yandex.ru Для работы с такой электронной почтой не требуются специальные почтовые программы, т.к можно воспользоваться любым браузером для загрузки отображения соответствующей web-страницы.

Задание №1.Научиться создавать электронный почтовый ящик.

1. Запустить на компьютере программу Internet Explorer. В навигационной строке ввести адрес любого российского сервера бесплатной электронной почты, например http://www.yandex.ru.(Рис.30)

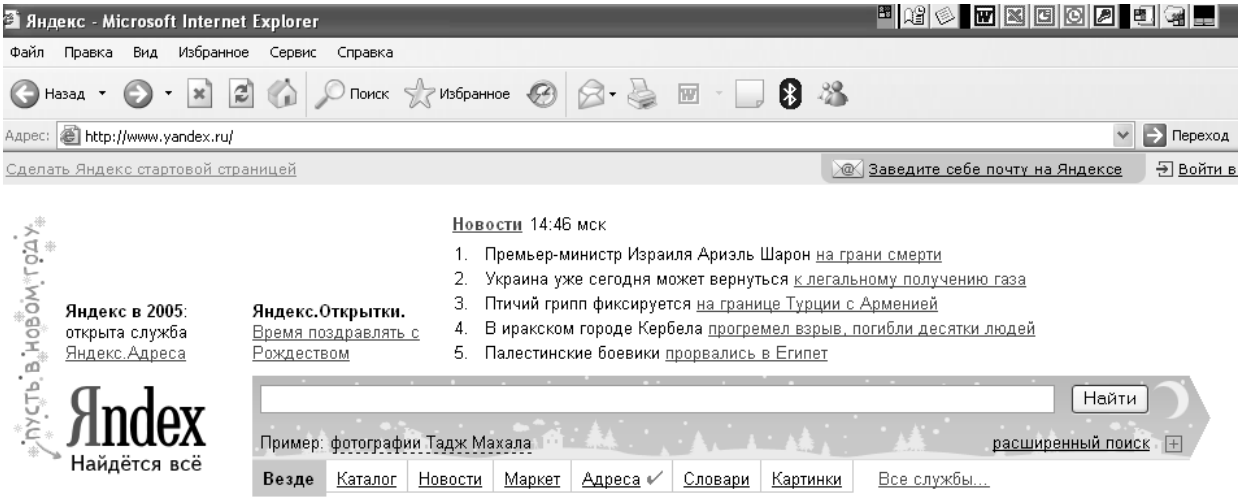


Рис.30 Бесплатный российский сервер электронной почты Яндекс.

2. Выбрать опцию «**Зарегистрироваться**». Для этого нажмите на гиперссылку «Заведи себе почту на Яндексе». В появившемся окне регистрации нового пользователя (Рис.31) необходимо создать имя почтового ящика. Оно состоит из двух частей – первая user_name ,

имя пользователя, вторая – server_name, имя почтового сервера, где происходит регистрация

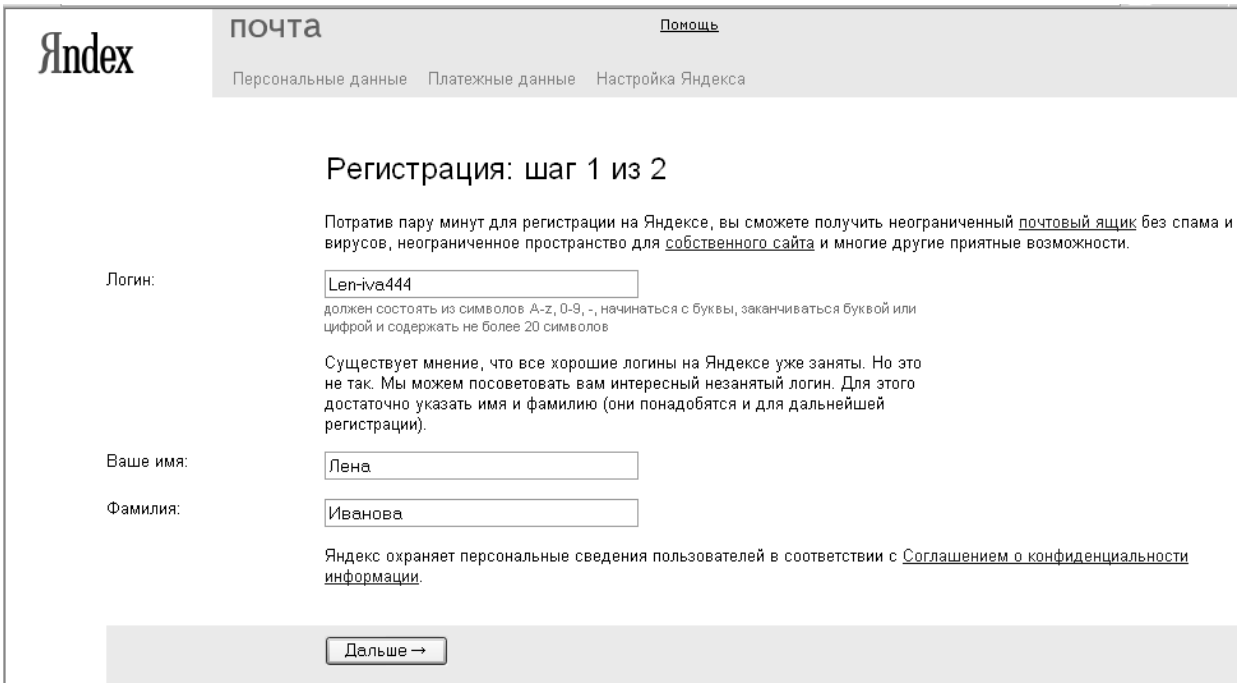


Рис.31. Первый шаг регистрации почтового ящика.

3. Придумайте для своего почтового ящика ***логин*** – это имя пользователя электронной почты, первая часть почтового адреса. Логин имеет произвольный характер, главное, чтобы он был оригинален для того доменного имени почтового сервера, где происходит регистрация.
4. Продумайте пароль для входа в Ваш электронный почтовый ящик. Он не должен быть слишком простым, в то же время лично для себя его необходимо запомнить. Оптимальный вариант пароля – сочетание буквенных и цифровых символов, понятное только лично самому пользователю.
5. Если все-таки пароль будет забыт Вами с течением времени по какой-либо причине, то необходимо создать ответ на контрольный вопрос (Рис.32). Ответ на такой вопрос будете знать только Вы.

Рис.32.Второй шаг регистрации почтового ящика

6. После заполнения всех необходимых полей формы регистрации выводится сообщение о том, что создан почтовый ящик для нового пользователя. При этом предлагается заполнить некоторые персональные данные (Рис.33). Теперь можно войти в почту или написать первое письмо.

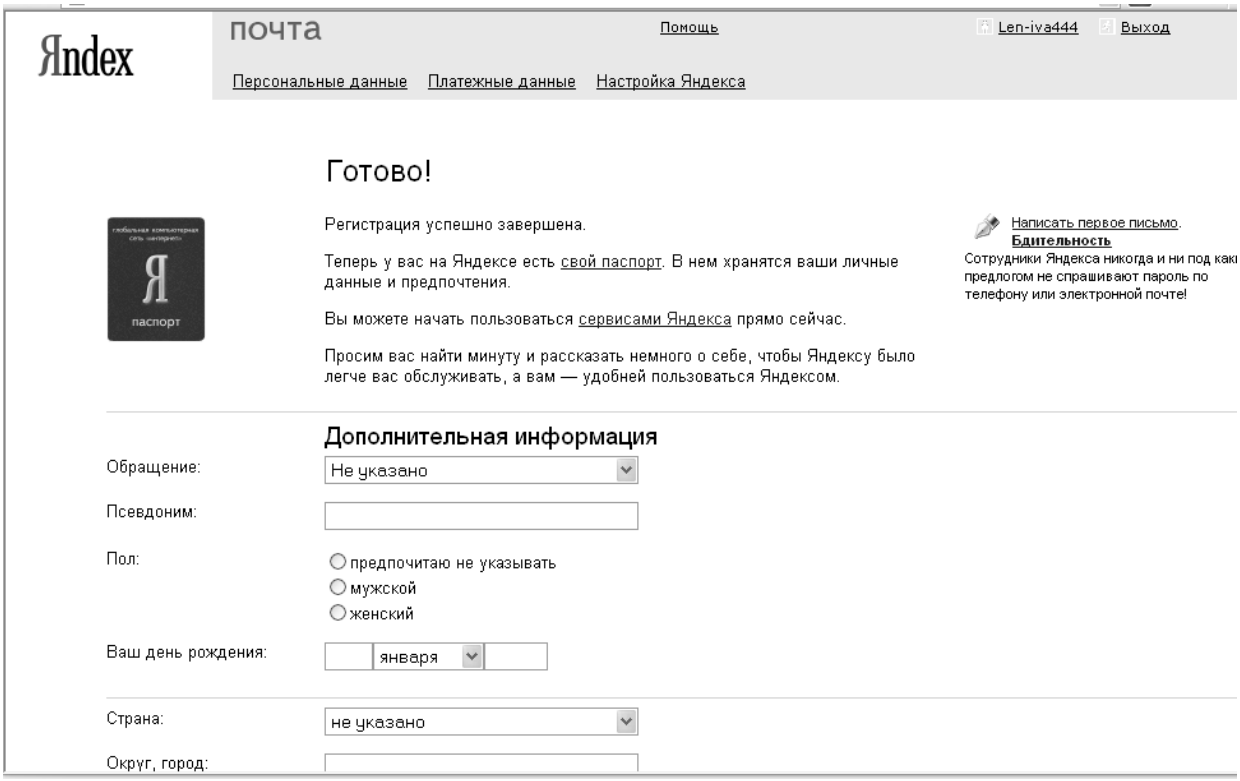


Рис. 33 . Результат создания электронного почтового ящика

Задание №2. Научиться создавать и отправлять электронные письма.

1. Для проверки писем в почтовом ящике, а так же для того, чтобы написать письмо зарегистрированные пользователи должны ввести свой идентификатор *(логин)* и пароль, после чего они могут войти в почтовую систему.
2. Выберите опцию ***Написать письмо*** (Рис.34.). Заполните графы **Кому** и **Тема.** Графа **Кому** заполняется либо вручную, либо из адресной книги. Ссылка Адресная книга позволяет создать виртуальную записную книжку с основными данными всех нужных Вам адресатов. **Тема** письма так же должна быть заполнена. Незаполненное окно **Тема** говорит либо о некомпетентности пишущего адресата, либо о его халатности или пренебрежении. Поэтому «электронный этикет» требует обязательного заполнения Темы письма. Окно **От Кого** заполняется автоматически. Текст письма может быть любым, однако не

помешает поверка грамматики и орфографии. (Кнопка **«Проверить…»**).

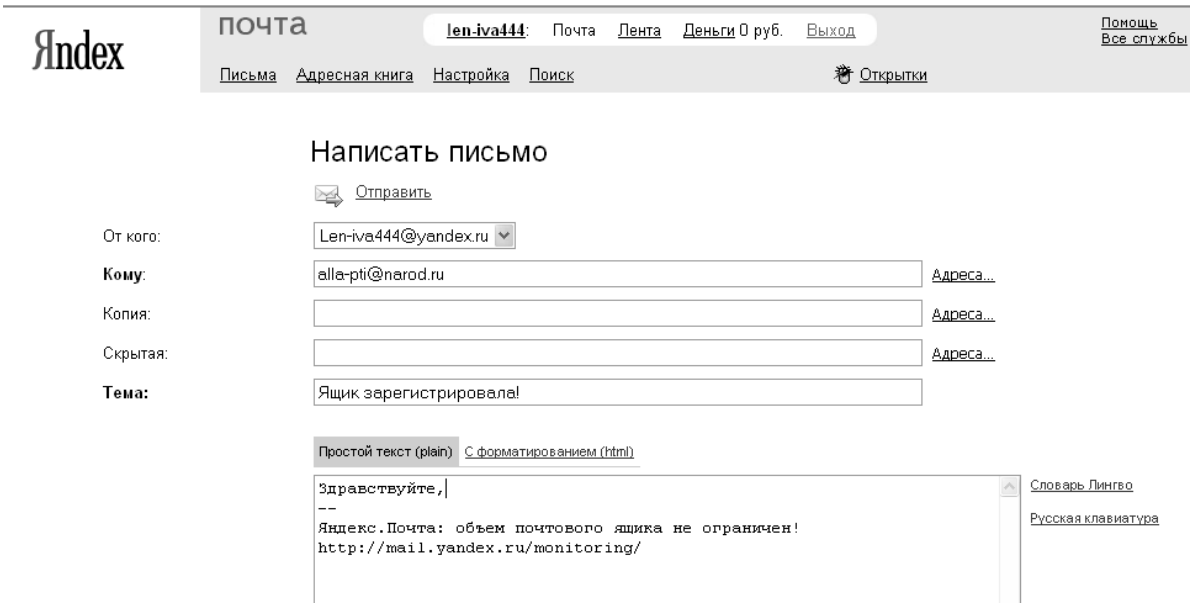


Рис.34.Пример написания электронного письма

3. Выберите Опцию (поставьте галочку) ***Сохранить копию письма при отправке***. Существенной особенностью Web-почты является то, что все сообщения постоянно хранятся на удаленном сервере, а не на локальном компьютере пользователя. Это часто бывает удобно, т.к. не всегда есть возможность отправлять необходимые файлы со своего компьютера. Если копия письма сохранена в ящике, то вложенные в отправленные письма файлы могут быть доступны для Вас с любого компьютера, имеющего подключение к Интернет.
4. Добавьте в письмо вложенный файл. Для этого нажмите кнопку «Обзор» у некоторых серверов эта опция называется «Вложить» или «Прикрепить». Обзор позволяет найти нужный файл на компьютере (Рис. 35). В данной работе вложите файл с отсканированным и распознанным документом (лаборатор-

ная работа №3) или файл с презентацией (лабораторная работа №4)

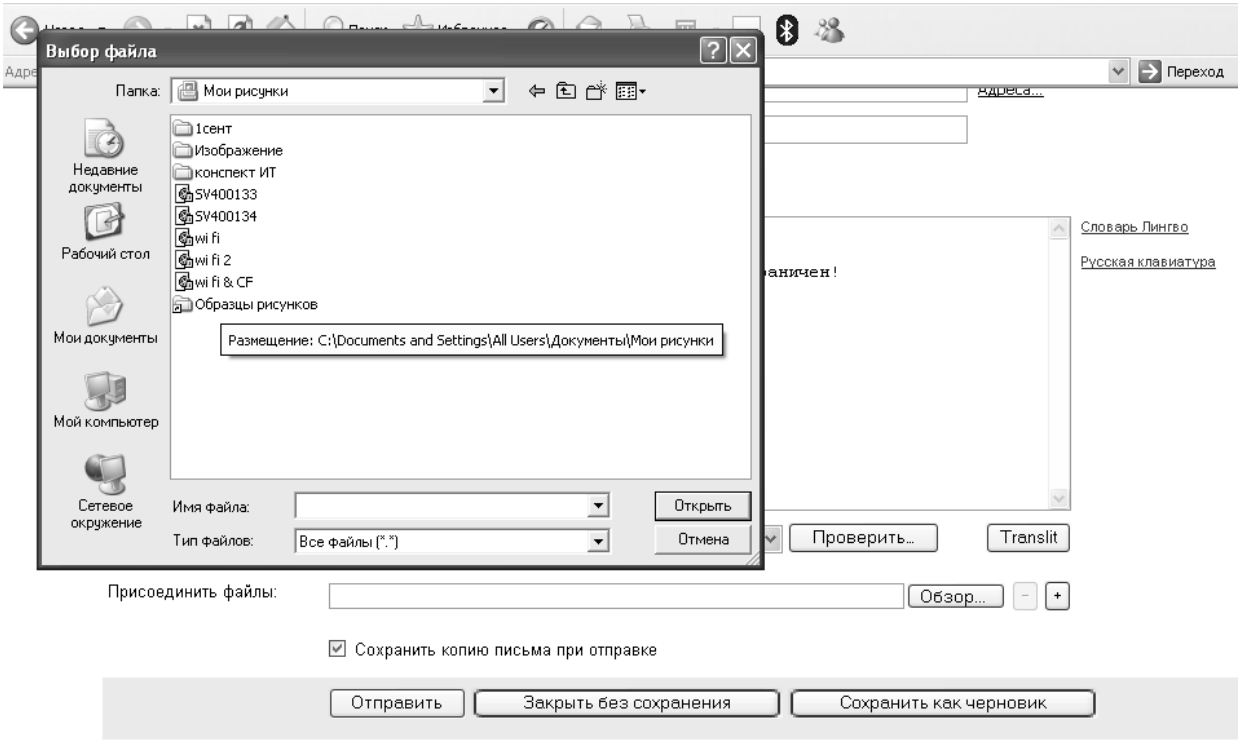


Рис.35.Пример написания электронного письма с прикрепленым (вложенным) файлом.

5. Отправьте готовое письмо с вложенным файлом. Это может занять некоторое время. Длительность отправки зависит от скорости линии, по которой имеется доступ в Интернет, а также от объема вложенного файла.
6. Дождитесь сообщения, что письмо отправлено (рис.36). Выберите опцию ***Вернуться в текущую папку***.

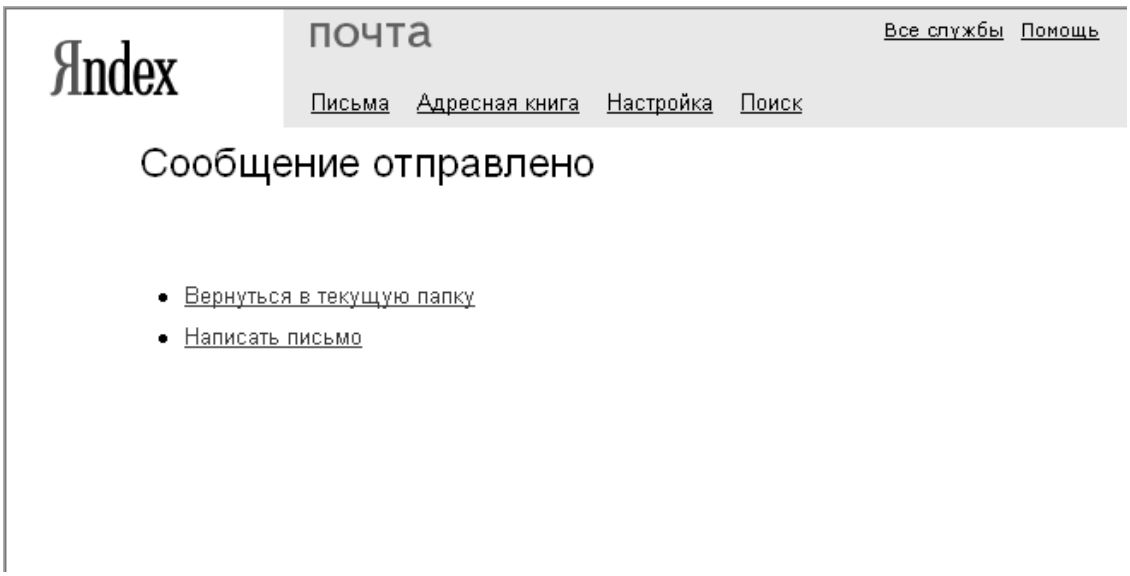


Рис. 36.Интерфейс программы после успешного отправления письма.

7. Просмотрите имеющуюся почту. Все пришедшие на Ваш адрес письма находятся в папке **Входящие** (рис.37).Если Вы работали в паре со соис сокурсником, то письма Вы могли отправить друг другу. Таким образом, в Вашем почтовом ящике появилось новое письмо .

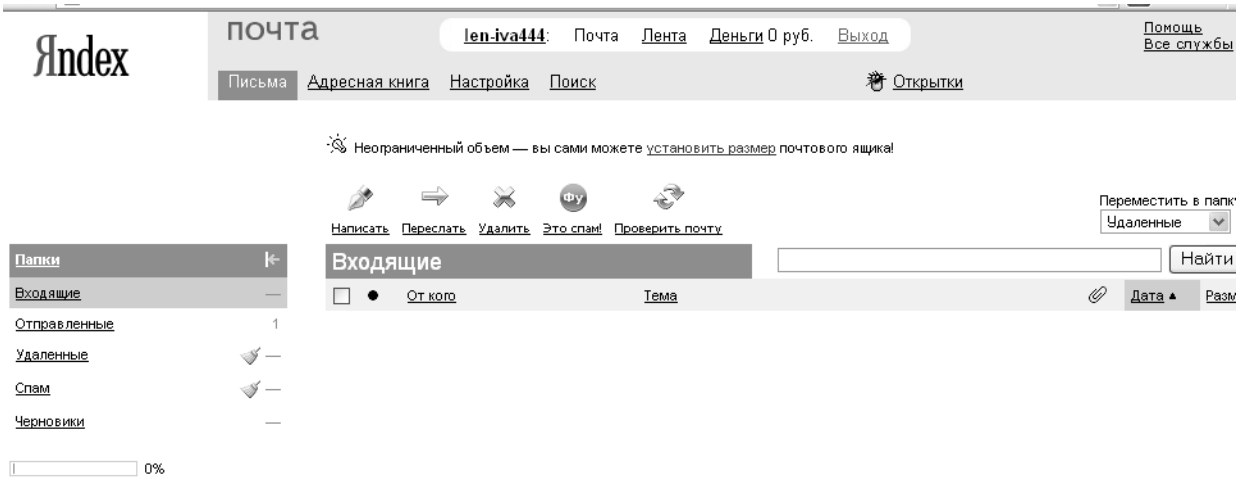


Рис. 37. Электронная** почта с Web-**Интерфейсом. Папка Входящие

Задание №3. Научиться настраивать почтовую программу на компьютере.

Для любых компьютерных платформ существует большое разнообразие почтовых программ. Среди них такие широко известные как Outlook Ex-

press, MS Outlook, The Bat! и другие. Суть работы такой программы в том, что адресат для получения письма должен соединиться с Интернетом и доставить почту со своего почтового ящика на удаленном почтовом сервере на свой локальный компьютер. При этом существует возможность доставки почты с нескольких ящиков, с нескольких почтовых серверов. Удобна также адресная книга и возможность одновременной рассылки писем нескольким адресатам.

В данной лабораторной работе предстоит настроить и использовать для работы с почтой программу MS Outlook. В первую очередь необходимо в соответствии с полученными в процессе регистрации почтового ящика данными (имя почтового ящика, пароль и др.) настроить почтовую программу. Создадим в почтовой программе Outlook Express учетную запись «Почта Интернета», при помощи которой можно будет отправлять и принимать электронную почту с созданного в первом задании почтового ящика.

1. Запустить программу MS Outlook. Появится окно, состоящее из 4 частей (Рис.38). В левой верхней части находится перечень папок с корреспонденцией и другие папки пользователя программы MS Outlook. Папки ***Входящие, Исходящие, Отправленные, Удаленные*** и ***Черновики*** аналогичны почтовой программе в web-интерфейсом. Пользователь может создавать собственные папки для хранения тематически сгруппированных сообщений. В папках могут храниться не только сообщения, но и файлы, созданные с помощью других приложений. В нижней левой части окна размещается список контактов, который предоставляет доступ к информации, хранящейся в ***Адресной книге*** (адреса электронной почты, телефоны и так далее).

Если программа не настроена и запускается впервые, то все папки из левого окна пустые.

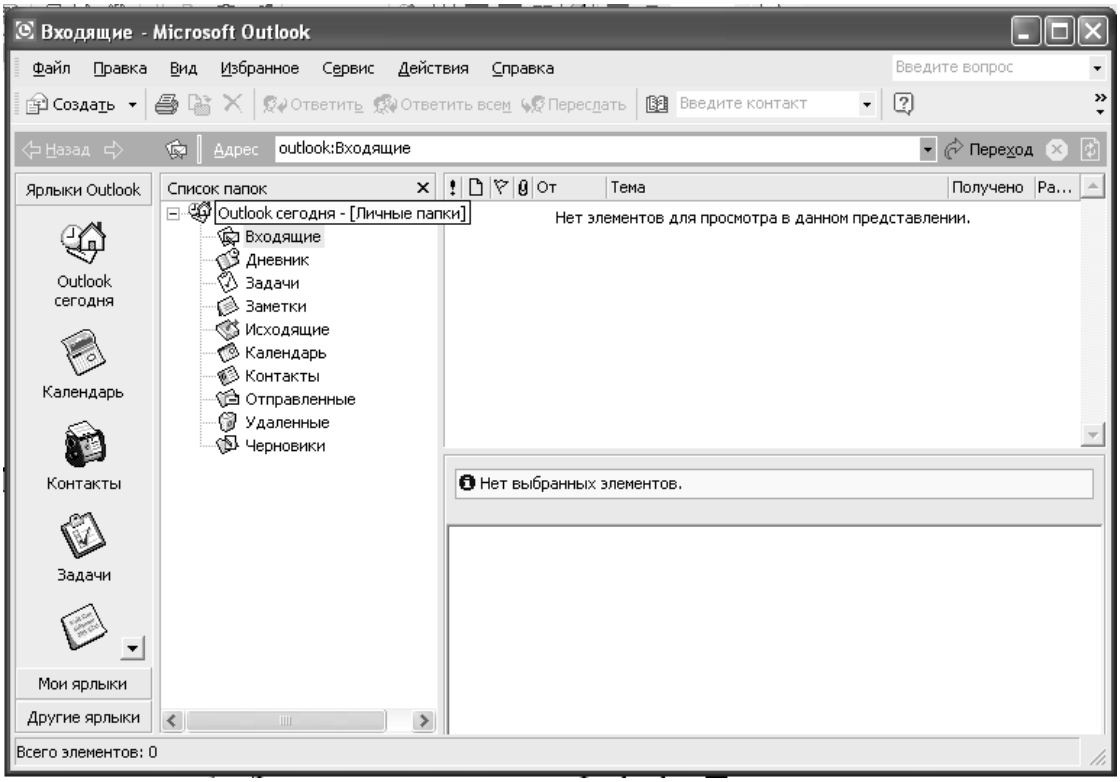


Рис.38. Окно почтовой программы MS Outlook

MS Outlook, будучи программой не только для работы с электронной почтой, но и планировщиком задач, имеет еще ***Календарь***, где пользователь разрабатывает планы и задачи, встречи, телефонные звонки на день или несколько дней вперед (Рис.39).

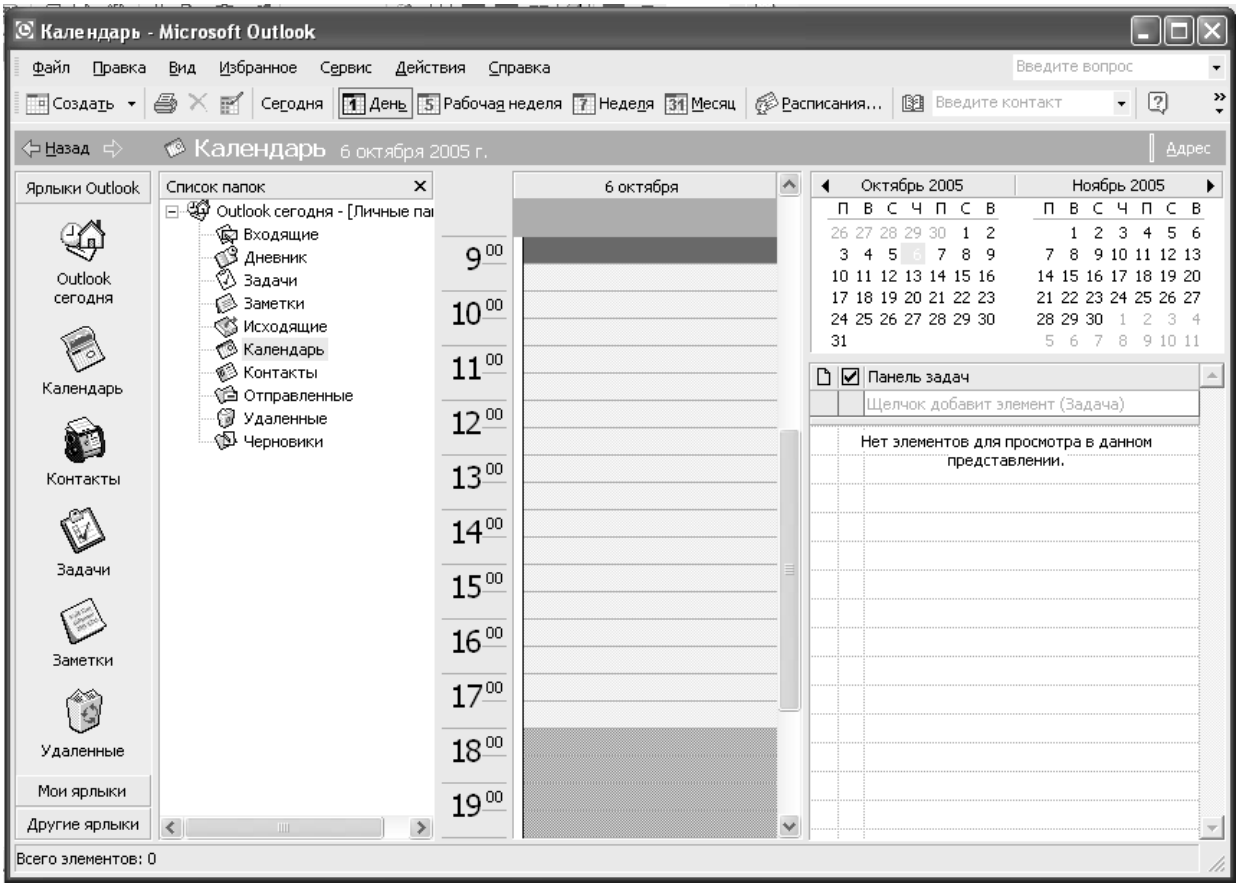


Рис.39. Календарь-планировщик задач в MS Outlook

2. Создать учетную запись в MS Outlook. Для этого ввести команду [Сервис – Учетные записи электронной почты]. Откроется диалоговая панель Учетные записи Интернета. Выбрать пункт Добавить новую учетную запись электронной почты. (Рис.40). Щелкнуть по кнопке Далее.

3. На появившейся диалоговой панели в поле ***Тип сервера*** для входящей почты: выберите POP3. Этот протокол наиболее часто используется для электронной почты.

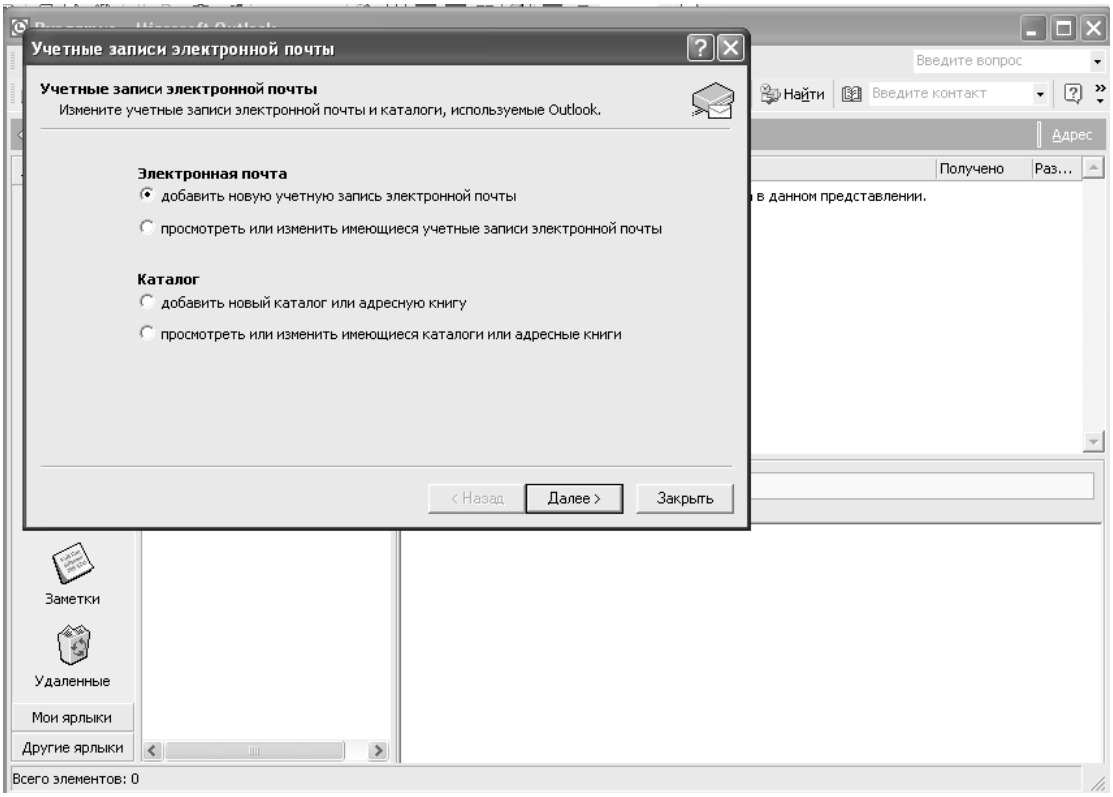


Рис. 40. Добавление новой учетной записи электронной почты

4. Теперь необходимо указать имя почтового ящика и пароль для входа на почтовый сервер (Рис.41). В поле ***Имя пользователя***: ввести имя (логин), которое вы указали при создании своего почтового адреса перед значком @. В поле ***Пароль***: необходимо указать тот пароль, который был получен при регистрации почтового ящика.

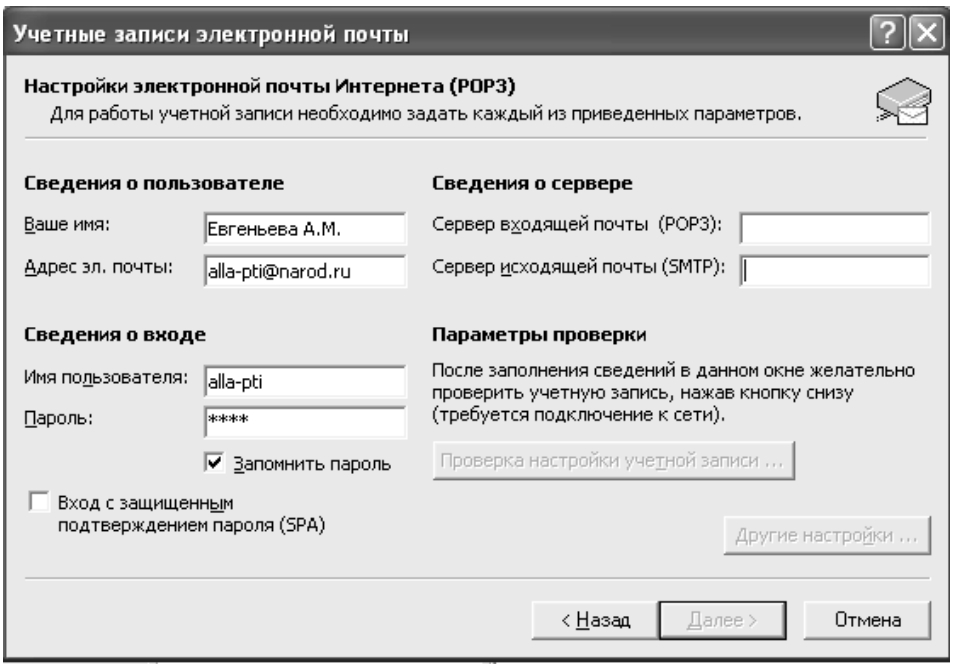


Рис. 42 .Создание учетной записи электронной почты

5. В поле ***Ваше имя***: указать имя, которое будет видеть человек, получивший от вас письмо. ***Адрес электронной почты***: указать тот адрес, который вы задали при регистрации подключения. Адрес следует указать целиком и именно в том виде, в котором вы его создали. Щелкнуть по кнопке Далее

6. В полях Сервер для входящей почты (POP3 или 1MAP4): и Сервер для исходящей почты (SMTP): необходимо указать имена серверов входящей и исходящей почты, которые сообщает провайдер при регистрации подключения (Рис.43). Щелкнуть по кнопке Далее

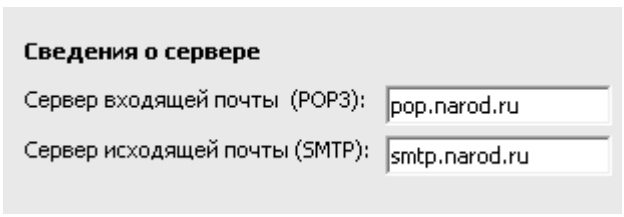


Рис. 43. Сведения о сервере на примере www.narod.ru.

7. Нажать кнопку ***Другие настройки*** и на появившейся панели указать способ соединения с Интернетом, выбрать тип модема и используемое соединение с Интернетом.

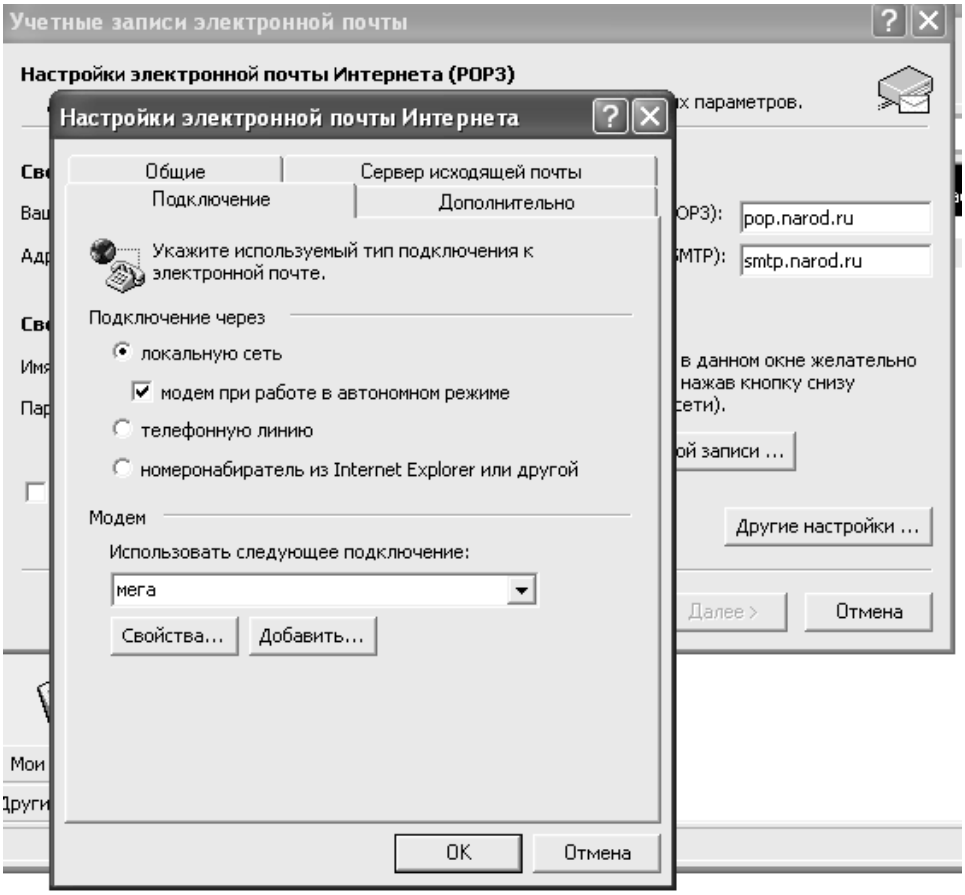


Рис.44. Настройки подключения к Интернет

8. Заданные выше параметры электронной почты объединяются вместе под одним именем — именем учетной записи. В поле Имя учетной записи почты сети Интернет: необходимо ввести имя для созданной учетной записи, например «ПочтаИнтернета». На рис.45 ***Новая учетная запись*** имеет имя АЛЛА.

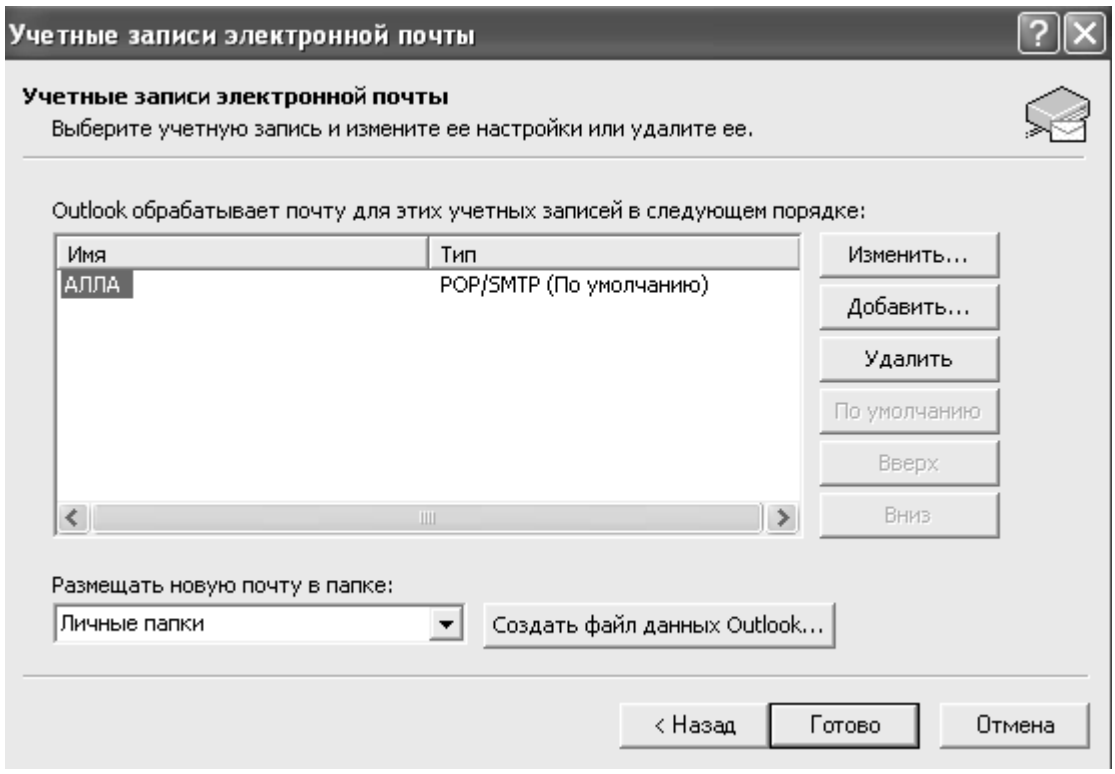


Рис. 45. Готовая учетная запись

Задание №4. Создание, отправка и получение сообщения с помощью почтовой программы.

1. Ввести команду [Сообщение-Создать]. В окне *Создать сообщение* в поле *Кому:* необходимо указать электронный адрес адресата, например: mip-kro@mail.ru В поле *Копии:* можно указать адреса получателей копии сообщения. В поле *Тема:* указывается тема сообщения, например «Проба».

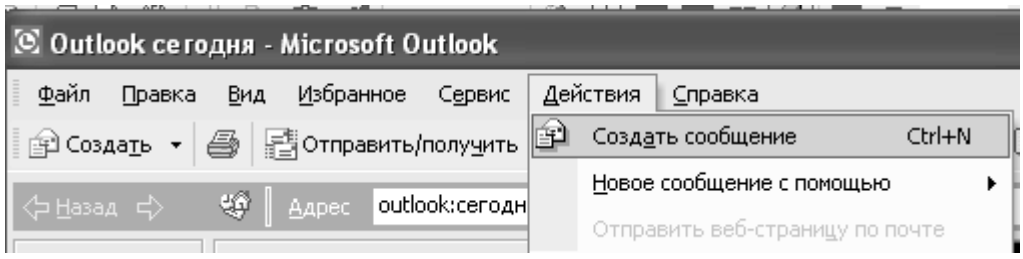


Рис. 46. Создать новое сообщение

2. В области, отведенной для сообщения, вводится текст сообщения, например «Пробное сообщение (кодировкаКОИ8-Р)». Достаточно важен вы-

бор правильной кодировки русских букв сообщения. При пользовании электронной почтой чаще всего используются кодировки Windows и КОИ8-Р

3. Выбор кодировки осуществить с помощью команды [Формат-Вид кодировки-Кириллица (КОИ8-Р)]. В сообщение можно вставлять файлы (текстовые, графические, звуковые и так далее).

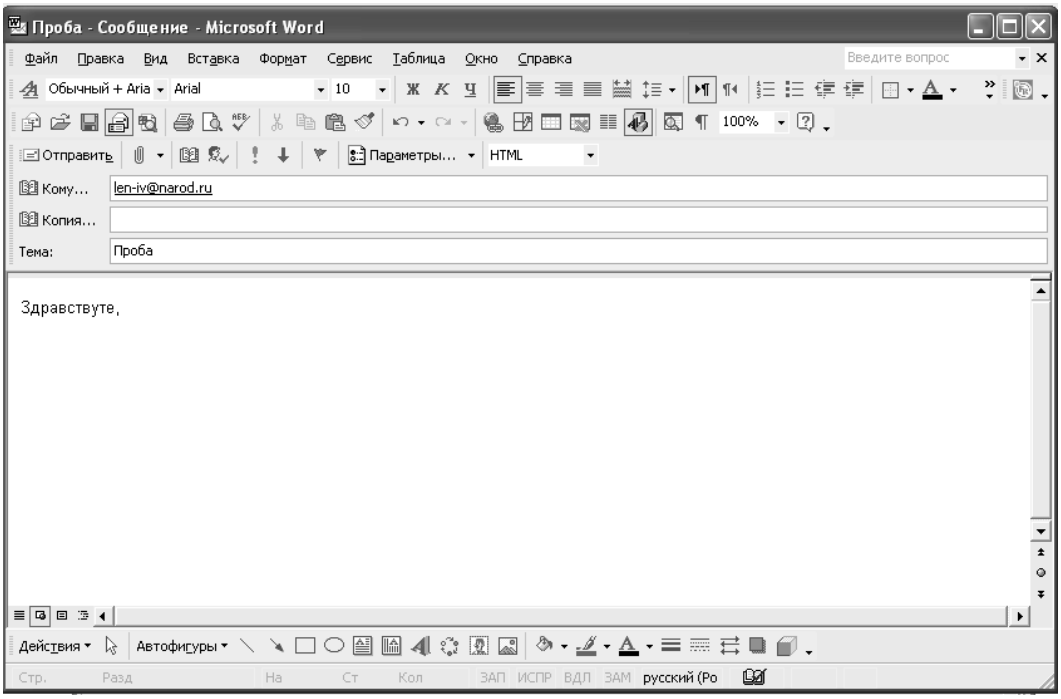


Рис. 47. Интерфейс почтовой программы MS Outlook: окно для написания письма

1. Для вставки файла в сообщение необходимо ввести команду [Вставка-Вложение файла...]. В появившемся окне *Вставка* необходимо выбрать требуемый файл, и он будет вложен в сообщение (Рис.48).

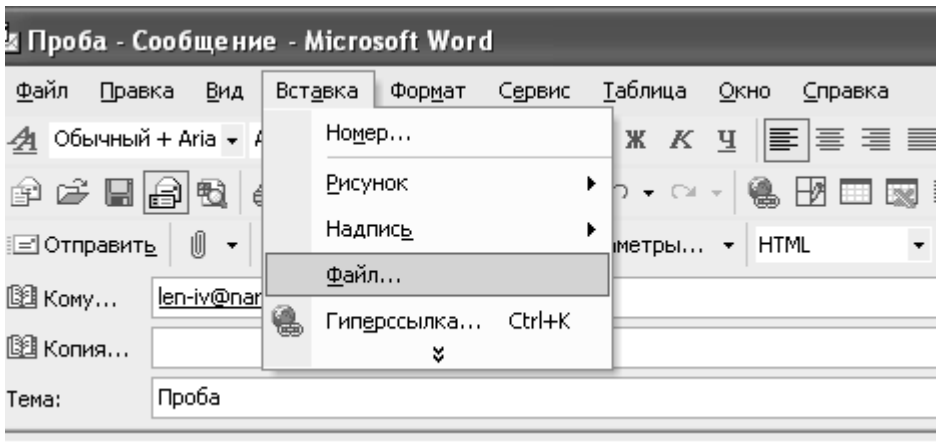


Рис. 48. Вложение файла в электронное письмо

2. Можно для вставки файла или документа воспользоваться на панели инструментов кнопкой с изображением скрепки. Любым способом вставим, например, в сообщение файл с презентацией из папки **Группа 287,** которая создана в лабораторной работе №3. Название вложенного файла появится в нижней части окна сообщения.
3. Если создание сообщения производилось в автономном режиме без подключения к Интернету, сообщение необходимо сохранить в папке Исходящие. После завершения работы над сообщением щелкнуть по кнопке ***Отправить***, сообщение будет помещено в папку ***Исходящие***
4. Для того чтобы отправить сообщение адресату, необходимо подключиться к Интернету. Щелкнуть по кнопке *Отправить почту.* Произойдет соединение с почтовым сервером, и все сообщения, находящиеся в папке *Исходящие* на локальном компьютере, будут доставлены на почтовый сервер. Одновременно отправленные сообщения будут перемещены на локальном компьютере в папку *Отправленные.* Почтовый сервер провайдера передаст сообщения в Интернет, и через некоторое время они будут доставлены на почтовые сервера получателей. В

данном случае пробное сообщение попадет в почтовый ящик len-iv@narod.ru. Для получения сообщения абонент должен соединиться с Интернетом и произвести операцию доставки почты с почтового сервера провайдера на свой локальный компьютер.

5. Щелкнуть по кнопке *Отправить/получить почту.* В процессе доставки почты сообщения, хранящиеся в почтовом ящике на почтовом сервере, будут переданы на локальный компьютер получателя и размещены в папке *Входящие.* В случае установки кодировки, отличной от использованной при создании сообщения, сообщение будет представлять собой полную абракадабру. В этом случае необходимо подобрать "кодировку с помощью команды [Формат-Вид кодировки...].

Самостоятельная работа

Научиться работать в телеконференциях.

В Приложении 12 подробно рассказано о назначении принципах работы служб Новости и Телеконференции. Для того чтобы иметь доступ к почтовому ящику какой-либо конференции, на нее необходимо «подписаться».

1.В окне почтовой программы Outlook Express выделить папку новостей (в данном случае *Конференции* и щелкнуть по кнопке *Группы новостей).*

2.Появится диалоговая панель ***Группы новостей,*** в которой высветятся названия десятков тысяч конференций. Для поиска интересующей тематики можно в поле ***Показать группы новостей****, содержащие:* задать символьный шаблон. Пусть, например, вас интересуют русскоязычные конференции, с большой долей вероятности их можно найти, если задать шаблон «rus». В окне откроется перечень конференций, в названии которых содержится этот шаблон. Следует выбрать некоторые из них (отметить с помощью мыши) и щелкнуть по кнопке ***Подписаться***. Произойдет загрузка содержимого этих конференций (писем участников) на локальный компьютер пользователя.

3.Вы выбрали три интересующие вас конференции, которые теперь существуют в форме папок, вложенных в папку конференций ***Конференции***. Выделите любую из них, например microsoft.public.ru.russian.outlookexpress (конференцию, посвященную проблемам работы в Outlook Express). В правом верхнем окне вы увидите ее содержание, то есть перечень присланных ранее сообщений.

4.Выделить интересующее вас сообщение, и в правом нижнем окне вы увидите содержание этого сообщения.

5.Ознакомившись с содержанием телеконференции, при желании можно послать в нее собственное сообщение, используя для создания бланка сообщения команду [***Сообщение-Ответить в группу новостей***].

ЛАБОРАТОРНАЯ РАБОТА № 21
ТУРБО ПАСКАЛЬ.
ЭКРАН ИНТЕГРИРОВАННОЙ СРЕДЫ

1. Вход в интегрированную среду.

Осуществляется посредством активизации файла bp.exe, который, как правило, записан в каталоге C:\BP\BIN (или D:\BP\BIN)

2. Структура экрана интегрированной среды.

1. основное меню – верхняя строка. Активизируется клавишей <F10> или щелчком мыши. С помощью клавиш управления курсором (<←>, <↑>, <→>, <↓>, <Home>, <End>) выбирается нужный пункт меню и раскрывается клавишей <Enter>. В раскрытом "падающем" меню выбирается нужный пункт и запускается также клавишей <Enter>. Можно заметить, что напротив наименований некоторых пунктов "падающего" меню записаны функциональные клавиши. Они позволяют выполнять нужные действия, не используя меню, что значительно повышает быстродействие при работе;

2. поле для размещения открываемых окон, в которых могут размещаться любые файлы, например с программами, исходными данными или результатами;
3. строка состояния – нижняя строка. Отражает состояние вычислительного процесса, представляет подсказку по назначению функциональных клавиш.

3. Выход из интегрированной среды.

Осуществляется с помощью меню File/Exit. Это означает, что необходимо в основном меню выбрать пункт File, войти в него с помощью клавиши <Enter>. Затем в "падающем" меню выбрать пункт Exit и запустить его на выполнение той же клавишей <Enter>.

Можно выйти из интегрированной среды и без помощи меню, нажав клавиши <Alt-X>.

В дальнейшем аналогичные действия не будут описываться столь подробно, а будут представляться в виде:

File/Exit или <Alt-X>

4. Работа в окне редактирования.

4.1. Открытие файла в окне редактирования.

Осуществляется через

File/Open/ имя файла или <F3>

При этом на экране открывается содержимое файла и указывается номер окна.

Замечания:

1) "имя файла" можно записать в строке «Name» или выбрать его из предлагаемого ниже списка, куда курсор перемещается клавишей <Tab> или <Enter>. Выбранный файл открывается клавишей <Enter>;

2) в предлагаемом списке выводятся файлы с расширением .pas. При желании можно вызвать любые файлы, указав соответствующий формат группы файлов в строке «Name».

4.2.Создание нового файла.

Осуществляется через File/New.

В результате открывается окно с именем "Noname" – "без имени". В этом окне можно записать текст файла, но для его сохранения ему нужно дать имя.

4.3. Задание имени файлу (или переименование файла)

Осуществляется через

File/Save as/ имя файла

Замечания:

1) если создается программа, в "имени файла" расширение можно не указывать. При этом автоматически файлу присваивается расширение .pas. Если файлу нужно другое расширение, то его необходимо указать в "имени файла";

2) при переименовании файла возможна его пересылка в другой каталог или диск, например на дискету. Для этого в "имени файла" записывается полное имя файла с указанием пути к нему, например: A:\MY\primer.pas (в этом случае будет создан файл primer.pas в каталоге MY на диске A:).

4.4. Сохранение файла.

Осуществляется по окончании работы с файлом и периодически во время работы с ним для предупреждения неприятностей, связанных с непредвиденным отключением электроэнергии, в результате чего вся информация в файле, записанная после последнего сохранения может быть утеряна.

Сохранение файла осуществляется через

File/Save или <F2>

5. Работа с блоками при редактировании текста.

Для того, чтобы скопировать, переместить или удалить часть текста его необходимо сначала выделить.

5.1. Выделение фрагмента.

Осуществляется клавишами перемещения курсора при нажатой клавише <Shift>:

<Shift- ←>, < Shift- ↑>, < Shift- →>, < Shift- ↓>, < Shift- Home>, <Shift- End>.

Снятие выделения осуществляется аналогично.

Для выделения блока текста с помощью мыши необходимо подвести указатель мыши к началу области выделения и, удерживая нажатой левую кнопку мыши, перетащить указатель мыши по выбранному фрагменту текста.

Для снятия выделения необходимо щелкнуть мышью по любому невыделенному участку текста.

5.2. Удаление фрагмента без восстановления.

Фрагмент выделить и нажать

Edit/Clear или < Ctrl - Delete>

5.3. Удаление фрагмента в буфер.

Фрагмент выделить и нажать

Edit/Cut или <Shift - Delete>

5.4. Копирование фрагмента в буфер.

Фрагмент выделить и нажать

Edit/Copy или <Ctrl - Insert>

5.5. Восстановление фрагмента из буфера.

Курсор поставить на место восстановления фрагмента и нажать

Edit/Paste или <Shift - Insert>

Для того чтобы скопировать фрагмент текста программы его необходимо сначала скопировать в буфер, затем установить курсор в нужное место программы и восстановить фрагмент из буфера. Для переноса фрагмента текста программы сначала необходимо удалить фрагмент в буфер, а затем восстановить фрагмент из буфера в нужном месте программы.

Замечания:

1. восстановление фрагмента из буфера возможно как после удаления фрагмента в буфер, так и после его копирования в буфер;
2. удаленный фрагмент восстановлению не подлежит;

3. в буфере хранится только последний внесенный туда фрагмент;
4. для восстановления любого фрагмента, внесенного ранее в буфер можно воспользоваться окном содержимого буфера, для чего следует нажать

Edit/Show clipboard

С содержимым окна буфера можно работать как с любым текстом, то есть можно редактировать, вставлять в текст программы и т.д.

6. Поиск с целью замены.

Этот режим используется в случае необходимости многократной замены одного набора символов на другой в тексте файла. Для этого необходимо:

1) войти в меню Search/Replace;

2) ввести в первую строку слово или сочетание символов для поиска;

3) с помощью клавиши <Tab> перевести курсор во вторую строку и ввести в нее новое сочетание символов, которым следует заменить старое;

4) с помощью клавиши <Tab> перевести курсор на Change all, что произведет глобальную указанную замену во всех вхождениях искомого набора символов от курсора до конца текста. При этом предлагается перед каждой заменой подтвердить согласие на нее клавишей <Enter> или отказ от нее.

Замечания:

Поиск и замена будет осуществляться, начиная с позиции курсора и до конца текста программы.

7. Работа с окнами.

Переход в заданное окно:

<Alt - № окна>

Закрытие окна редактирования:

Window/Close или <Alt - F3> или пиктограмма для мыши [■]

Открытие пользовательского экрана (на этот экран обычно выводятся результаты вычислений программы): <Alt - F5>. Для возвращения в окно редактирования можно нажать любую клавишу.

Размещение окон встык (используется при работе с несколькими файлами): Window/Tile.

Размещение окон с перекрытием друг друга:

Window/Cascade.

Удаление (закрытие) всех окон:

Window/Close all.

Изменение размеров окна:

Window /Size/Move/<Shift - ←, ↑, →, ↓> или <Ctrl - F5>

Для фиксации выбранного размера служит клавиша <Enter>.

Перемещение окна:

« Window »/«Size/Move»/<←, ↑, →, ↓> или <Ctrl - F5>

Для фиксации выбранного положения окна служит клавиша <Enter>.

Распахивание окна во весь экран:

Window/Zoom или <F5>

Активизация следующего окна:

Window/Next или <F6>

Активизация предыдущего окна:

Window/Previous или <Shift - F6>

Открытие списка окон:

Window/List или <Alt - 0>

8. Запуск программы на выполнение.

Run/Run или <Ctrl - F9>

После запуска программы на выполнение результаты работы программы выводятся на экран пользователя <Alt - F5>. Для возвращения в окно редактирования можно нажать любую клавишу.

ЗАДАНИЕ

1. Войти в интегрированную среду Турбо Паскаль.

2. Создать файл с именем XYZ781-1,

где символы X, Y, Z представляют Ваши собственные инициалы в латинском алфавите и их нужно заменить на соответствующие буквы;

781 - означает номер группы (ввести свой номер группы);

1 - означает порядковый номер созданного Вами файла.

3. В содержимое файла записать программу:

```
PROGRAM  integr_sreda;
  BEGIN
     writeln (' Это моя первая программа');
     writeln (' Тема: "Интегрированная среда" ');
     writeln (' Я студент первого курса АмГУ ');
END.
```

4. Сохранить созданную программу.

5. Запустить программу на выполнение.

6. Посмотреть на экране и записать в тетрадь результат выполнения программы.

7. Средствами работы с блоком скопировать предпоследнюю строчку программы 4 раза перед словом END.

8. Средствами глобальной замены в последних трех скопированных строчках заменить "АмГУ" на "МГУ".

9. Запустить программу. Посмотреть результат ее выполнения.

10. Средствами работы с блоком перенести строчку "writeln (' Это моя первая программа');" в положение перед словом END.

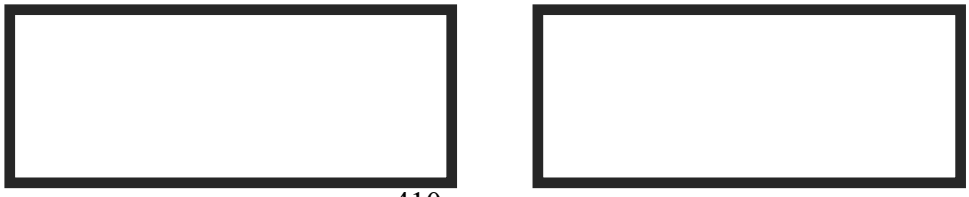
11. Средствами работы с блоком удалить строчку "writeln (' Тема: "Интегрированная среда" ');" в буфер.

12. Открыть окно содержимого буфера и скопировать строку " ' Тема: "Интегрированная среда" " из этого окна в текст предпоследней строки программы в любое место между ( ).

14. Расположить окна с программой и содержимым буфера встык.

15. Вернуться к каскадному расположению окон.

16. Изменить размеры и расположение окон для представления их на экране в виде:



17. Распахнуть окно с программой во весь экран.

18. Вернуться к прежнему расположению окон. Проверить различные варианты перемещения между окнами. Сохранить программу.

19. Закрыть окна.

20. Открыть созданный файл. Внести произвольные исправления в текст и запустить программу на выполнение.

21. Сделать копию Вашего файла с именем XYZ781-2.

22. Выйти из интегрированной среды Турбо Паскаль. Найти созданные файлы. Скопировать файлы на дискету.

Самостоятельная работа

1.Изучить меню интегрированной среды Турбо Паскаль.

2.Ознакомиться с приемами редактирования текста программы.

3.Самостоятельно создать программу вывода на экран текста. Сохранить ее и запустить на выполнение.

4.Расположить окно с созданной программой различным образом, открыть данную программу в нескольких окнах.

ЛАБОРАТОРНАЯ РАБОТА № 22

ТУРБО ПАСКАЛЬ. СТРУКТУРА ПРОГРАММЫ. КОМПИЛЯЦИЯ ПРОГРАММЫ

1. Оформление текста программы.

6. компилятор воспринимает строки до 126 символов;

7. клавиша <Enter> переводит курсор на новую строку в положение начала предыдущей строки;

8. группа операторов, выполняющих законченное действие, выделяется пустыми строками до и после нее;

9. идентификаторы могут включать строчные и заглавные буквы английского алфавита, цифры, знак подчеркивания "_";

10. для семантически связанных групп операторов используются комментарии в фигурных скобках {}.

11. не следует давать локальным и глобальным переменным одинаковые имена.

2. Перемещение курсора по экрану.

12. на одно слово вправо / влево: <Ctrl - ←>, < Ctrl - →>;

13. на один экран вверх / вниз: <Page Up>, <Page Down>;

14. скроллинг на одну строку вверх / вниз: <Ctrl - W>, < Ctrl - Z>;

15. к началу / концу строки: <Home>, <End>;

16. на первую / последнюю строку окна: < Ctrl - Home>, < Ctrl - End>;

17. к началу / концу текста: < Ctrl - Page Up>, < Ctrl - Page Down>.

3. Компиляция программы.

3.1. Компиляция предусматривает:

18. проверку программы на правильный синтаксис с указанием места и типа ошибки;

19. преобразование файла с языка высокого уровня в машинно-ориентированный, доступный для прочтения и выполнения машиной; если скомпилированный файл образуется на диске, то его расширение **.pas** заменяется на **.exe**.

3.2. Местоположение скомпилированного файла.

Зависит от установка в меню Compile/Destination. Если в указанном пункте установлено «Destination Memory», то скомпилированный файл (файл с расширением .exe) будет храниться только в памяти компьютера. Если установлено «Destination Disk», то файл можно найти в одном из каталогов диска.

Замечание: Если в меню Compile нет опции Destination, то скомпилированный файл (файл с расширением .exe) автоматически образуется в одном из каталогов диска.

Местоположение скомпилированного файла устанавливается в первой строке открытого меню Options/Directories.

Для сохранение введенной установки местоположения скомпилированных файлов используется меню Options/ Save BP.tp.

3.3. Компиляция программы осуществляется:

20. при запуске программы на выполнение;

21. через меню Compile/Compile или<Alt - F9>.

3.4. Действия для получения скомпилированной программы в виде .exe - файла:

22. задать расположение **.exe -** файла на диске (если такая установка есть);

23. установить местоположение **.exe -** файла, войдя в меню Options/ Directories (см. п. 3.2.);

24. скомпилировать файл (см. п. 3.3.);

25. выйти из среды Турбо Паскаль и найти в установленном каталоге **.exe –** файл;

26. запустить **.exe –** файл на выполнение из операционной системы MS DOS или WINDOWS (нажав клавишу <Enter> на **.exe –** файле).

Замечание: **.exe -** файл нужен для возможности его запуска из операционной системы MS DOS или WINDOWS, не заходя в интегрированную среду Турбо Паскаль.

ЗАДАНИЕ

1. Изучить пример программы вычисления значения переменной Z

$$Z = a \cdot x^2 + \frac{b}{\sqrt{\pi \cdot y}} - 18.01$$

при произвольно заданных в процессе выполнения программы значениях переменных X и Y и заданных значениях констант: a = 100, b = 3,2.

Комментарии по программе	Пример программы
Заголовок программы. Имя должно быть лат.буквами	PROGRAM struc_program;
Раздел констант	CONST a=100; b=3.2;
Раздел переменных. X и Z зада-	VAR x, z:real;

ны вещественными, Y – целое	y: integer;
Начало раздела операторов	BEGIN {Начало работы программы}
Операторный ввод данных в диалоговом режиме	write (' Введите любое значение X = '); readln (x); write (' Введите целое значение Y = '); readln (y);
Вычисления	z:=a* sqr(x)+b/sqrt(y* PI) —18.01;
Вывод результатов:	writeln (' Результаты вычислений ');
в экспоненциальной форме	writeln ('Z=' ,Z);
в формате 5:2 (5 знаков, из них 2 – после запятой)	writeln ('Z=' ,Z:5:2);
без знаков после запятой	writeln ('Z=' ,Z:5:0);
в экспоненциальной форме	writeln ('Z=' ,Z:3);
Конец раздела операторов	END. {Конец работы программы}

2. Записать программу в файл XYZ781-3. Запустить программу и получить результаты. Проверить работу программы при различных исходных данных.

3. Откомпилировать программу с целью получения **.exe -** файла. Найти его на диске и запустить программу на выполнение из операционной системы MS DOS или WINDOWS.

4. В тетрадь записать программу с комментариями и отчет по ее работе (вид пользовательского экрана после выполнения программы).

Самостоятельная работа

1.Самостоятельно написать программу расчета корней квадратного уравнения.

2.Вывести значения корней квадратного уравнения в различных вариантах.

3.Создать **.exe –** файл в любом каталоге на жестком диске, запустить данную программу из операционной среды WINDOWS.

ЛАБОРАТОРНАЯ РАБОТА № 23
ТУРБО ПАСКАЛЬ.
ОПЕРАТОРНЫЙ ВВОД - ВЫВОД ДАННЫХ

ЗАДАНИЕ

1. Разработать программу по индивидуальному заданию записать ее в файл XYZ781-4.

Вариант1

Определить пылепроницаемость швов ткани $П_{ш}$, г/м^2:

$$П_{ш} = П – П_{т},$$

где $П_{т}$ – пылепроницаемость ткани г/м^2;

П – пылепроницаемость ткани со швом, г/м^2.

$$П_{т} = (M_0 – M)/S; \quad П = (M_0' – M ')/S ',$$

где M_0 – масса пробы ткани до испытания. $M_0 = 6,6$ г;

M – масса пробы ткани после наполнения пылью. M = 7,5 г;

M_0' – масса пробы ткани со швом до испытания. $M_0' = 6,9$ г;

M ' – масса пробы со швом после наполнения пылью. M ' = 8,01 г;

S , S ' – площади проб, соответственно до и после испытания, м^2:

$$S = S ' = a \cdot b \cdot 10^{-6}$$

где a, b – размеры образцов: a = 126 мм, b = 176 мм.

В программе должны быть учтены следующие условия:

- ввод данных осуществить с помощью операторов в диалоговом режиме;
- вывод результатов должен быть представлен в удобном для прочтения виде с комментариями о наименовании и единицах измерения получаемых величин;
- в качестве результатов должны быть выведены не менее трех переменных;
- в выводе результатов указать, по каким исходным данным получены результаты.

2. Получить **.exe -** файл. Найти его местоположение на диске. Убедиться в работоспособности программы из операционной системы MS DOS или WINDOWS.

Самостоятельная работа

1. В тетрадь записать:

- программу;
- комментарии по программе аналогично примеру в лабораторной работе № 22;
- результаты выполнения программы;
- местоположение **.exe -** файла;
- объем памяти, занимаемой файлами с расширением **.exe** и **.pas**.

2. Файл программы записать на дискету.

3. Самостоятельно внести изменения в задание и отразить эти изменения в созданной программе.

ЛАБОРАТОРНАЯ РАБОТА № 24
ТУРБО ПАСКАЛЬ. ФАЙЛОВЫЙ ВВОД ДАННЫХ

Считывание данных из файла обеспечивается включением в программу следующих элементов.

1. В разделе переменных следует объявить две переменные:

- файловую типа **text** для организации посредством нее связи между программой и файлом исходных данных;
- типа **string** для хранения имени файла исходных данных. Данная переменная может отсутствовать, если в тексте программы указано конкретное имя файла исходных данных.

Пример:

VAR n1: string; f1:text;

2. В разделе операторов следует организовать:

- ввод имени файла в диалоговом режиме и запись его в переменную типа **string.**

Пример:

```
write (' Введите имя файла исходных данных');
readln (n1);
```

На возникающий в процессе выполнения программы запрос необходимо ввести имя файла исходных данных, например **ttt.dat**.

Замечание: если программа работает только с одним файлом, то данный запрос не нужен.

- связь между файловой переменной и файлом исходных данных с помощью процедуры **assign (f1,n1).**

Замечание: если программа работает только с одним файлом, то вместо переменной **n1** можно записать конкретное имя фала, например **assign (f1, ttt.dat).**

- открытие файла для считывания из него данных с помощью процедуры **reset(f1);**
- считывание данных из файла с помощью оператора ввода данных, например **readln(f1,a,b,c)**. Данный оператор считает из файла с помощью файловой переменной **f1** значения переменных **a,b,c**, поэтому он должен стоять до первого использования считываемых переменных.

Замечания:

1) на первом месте в операторе **readln** ставится имя файловой переменной, на втором – перечисляются идентификаторы тех переменных, чьи значения нужно считать из файла;

2) считываемые из файла переменные в операторе **readln** перечисляются в том порядке, в каком записаны их значения в файле исходных данных.

3) если исходные данные считываются из файла, то их ввод в диалоговом режиме с клавиатуры не нужен;

4) вид оператора **readln** зависит от расположения данных в файле исходных данных. Если данные записаны в одну строку, то их считывание возможно одним оператором **readln**, если в несколько строк – то и операторов должно быть несколько.

5) вид файла исходных данных для оператора **readln(f1,a,b,c):**

23 2 1.3

При этом после выполнения оператора **readln** в переменную **a** запишется значение 23, в переменную **b** – 2 , в переменную **c** – 1.3.

- закрытие файла исходных данных с помощью процедуры **close (f1),** которую можно поставить в программе после считывания всех данных из файла или, как чаще всего и делают, перед закрытием программы словом **END**.

ЗАДАНИЕ

Отредактировать программу из лабораторной работы № 23, заменив ввод данных в диалоговом режиме с клавиатуры на ввод из файла. Файл исходных данных назвать **date_4.dat**.

Самостоятельная работа

1.Создать новый файл исходных данных, в котором расположить исходные данные в столбик.

2. Изменить созданную программу в соответствие с расположением данных в файле исходных данных.

ЛАБОРАТОРНАЯ РАБОТА № 25

ТУРБО ПАСКАЛЬ. ВЫВОД РЕЗУЛЬТАТОВ В РАЗЛИЧНЫХ ВАРИАНТАХ

Для вывода результатов используются операторы **write(), writeln(),**

где в () указываются данные с различными форматами вывода.

ЗАДАНИЕ

Отредактировать программу из лабораторной работы №24, организовав вывод результатов в программе в следующих вариантах:

1. число

   число

   число

2. число число число
3. x= число y= число z= число
4. x= число*** y= число*** z= число
5. x= число

****

y= число

****

z= число

6. x= число, ед.изм., y= число, ед.изм., z= число, ед.изм.
7. Наименование x= число, ед.изм.,

Наименование y= число, ед.изм.,

Наименование z= число, ед.изм.

8.

Наименование	Единицы измерения	Обозначение	Значение
Скорость	м/с	x	3.05

9. В формате с точностью до 3-х знаков после запятой:

x= __число, y= __число, z=__ число

10. В экспоненциальной форме

x= число y= число z= число

11. число

    число

число

12. число_ _ _ _ _ _ число

число

Самостоятельная работа

1. Вывести результаты и исходные данные с комментарием в столбик с точностью до 5-го знака после запятой.

2. Добавить в таблицу вывод исходных данных.

3. Самостоятельно разработать несколько вариантов вывода.

ЛАБОРАТОРНАЯ РАБОТА № 26

ТУРБО ПАСКАЛЬ. ФАЙЛОВЫЙ ВЫВОД ДАННЫХ

Запись результатов вычисления программы в файл обеспечивается введением в программу следующих элементов.

1. В разделе переменных следует объявить две переменные:

- файловую типа **text** для организации посредством нее связи между программой и файлом результатов;
- типа **string** для хранения имени файла результатов. Данная переменная может отсутствовать, если в тексте программы указано конкретное имя файла результатов.

Пример:

VAR n2: string; f2:text;

2. В разделе операторов следует организовать:

- ввод имени файла в диалоговом режиме и запись его в переменную типа **string.**

```
writeln (' Введите имя файла результатов');
readln (n2);
```

На возникающий в процессе выполнения программы запрос необходимо ввести имя файла результатов, например **ttt.dat**.

Замечание:

1. Если файла с таким именем нет, то он создается;

2. Если файл существует, то выдается запрос «заменить ли его содержимое новым»? Ответить – «да».

- связь между файловой переменной и файлом результатов с помощью процедуры **assign (f2,n2).**

Замечание: Если программа работает только с одним файлом, то вместо переменной **n2** нужно записать конкретное имя фала, например **assign (f2, ttt.dat).**

- открытие файла результатов для записи в него данных с помощью процедуры **rewrite(f2);**

- в операторах вывода результатов **write, writeln** в скобках на первом месте пишется имя файловой переменной **f2**, связанной с файлом результатов. Остальная выводимая информация выводится аналогично выводу на экран:

writeln (f2, 'x1= ', x1:5:2);

- закрытие файла результатов с помощью процедуры **close (f2),** которую можно поставить после окончания вывода результатов в файл. Лучше его поставить перед словом **END**.

ЗАДАНИЕ

Отредактировать программу из лабораторной работы № 25, заменив вывод результатов с экрана на вывод в файл. Файл результатов назвать **date_5.dat**.

Самостоятельная работа

Создать несколько файлов для вывода. В первый файл поместить исходные данные, во второй – результаты, в третий – таблицу с результатами и исходными данными.

ЛАБОРАТОРНАЯ РАБОТА № 27

ТУРБО ПАСКАЛЬ. РАБОТА С СИМВОЛЬНЫМИ И СТРОКОВЫМИ ДАННЫМИ

Вариант 1

x=′ *Скорость движения V = 7.8 м/с′*

y= *′ Особо опасная зона!!!′*

z= *′ без′*

k= *′ОСТОРОЖНО !!!′*

ЗАДАНИЕ

1. Определить и вывести на экран длину всех строковых констант x, y, z, k.

2. Организовать получение строковых констант:

а) a1, которой присваивается значение скорости;

б) a2, соответствующей строке *′ Особо безопасная зона!!!′ ;*

в) a3, соответствующей строке *′ОСТОРОЖНО !!! Особо безопасная зона!!!′* ;

г) a4, соответствующей строке y, в которой между первым и вторым словом вставлены три восклицательных знака;

д) a5, соответствующей *′ОСТОРОЖНО !!! зона!!!′;*

е) Определить и вывести на экран в одну строку через запятую код каждого символа из строковой константы z ;

ж) Вывести в одну строку на экран символы с кодами:

140, 128, 131, 136, 159, 1, 1, 1

Замечание: ввод и вывод данных организовать через файлы.

Самостоятельная работа

1. Определить и вывести в файл длину всех полученных строковых данных a1, a2, a3, a4, a5.

2. Самостоятельно создать строковые данные a6, a7, a8 из имеющихся исходных данных. Вывести полученные строковые данные в новый файл результатов.

ЛАБОРАТОРНАЯ РАБОТА № 28

ТУРБО ПАСКАЛЬ. СТАНДАРТНЫЕ ФУНКЦИИ

ЗАДАНИЕ

•Найти значение функции z(t*)* при заданном значении t. Для этого записать выражение z(t*)*, используя стандартные функции Турбо Паскаля.

•Определить значения указанных в задании стандартных функций, обозначив их z1, z2, z3, z4.

•Определить принадлежит ли заштрихованной области точка с координатами (x,y). Для этого записать выражение, определяющее принадлежность точки к заштрихованной области.

•Результаты вывести на экран в виде:

Программу составил Иванов И.И.

При t=-1.57 z=-15.547

z1=-15 z2=-16 z3= FALSE z4= FALSE

Точка (0.5;1.2) лежит в заштрихованной области – FALSE.

Пример задания

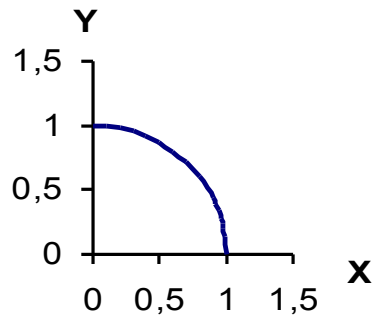
1. $z(t) = 3^{-t+1} \cdot \sin t$ при t = -1,57

2. z1 – целая часть от числа *z*, полученная путем отбрасывания дробной части.

   z2 - целое значение, полученное из числа *z* путем его округления

   z3 - определяет положительно ли *z*.

   z4 – проверяет нечетность числа z2.



3. (0,5;1,2)

Программа по индивидуальному заданию:

```
PROGRAM PRIMER;
VAR     x, y, z, t: real; z1, z2: integer; z3, z4:boolean;
BEGIN
    write ('Введите t=');                  {Ввод в диалоговом режиме значения
    readln (t);                            переменной t}
    write ('Введите координаты точки x='); readln (x);    {Ввод координат точки}
    write ("Введите координаты точки y='); readln (y);
    z :=exp((-t+1)*ln(3))*sin(t);          {Расчет значения функции z}
    writeln ('Программу составил Иванов И.И.');
    writeln ('при t=', t:5:2, '  z=', z:6:3);          {Вывод значения функции z при
                                                       заданном значении t}
    z1:=trunc(z); z2:=round(z);                        {Расчет значений  z1, z2, z3, z4}
    z3:=z>0; z4:=odd(z2);
    writeln ('z1=', z1:4, ' z2=', z2:4, '  z3=', z3, '  z4=',z4); {Вывод значений z1, z2,
                                                                   z3, z4}
write ('Точка (',x:3:1,';',y:3:1,')  лежит  в  заштрихованной области-   ');
writeln ((x>0) and (y>0) and (sqr (x)+sqr(y)<=1);
END.
```

Выражение различных функций через стандартные функции языка Паскаль

Функция	Функция на языке Паскаль
$y = x^z$	$y := \exp(z * \ln(x))$
$y = \arcsin x = arctg \frac{x}{\sqrt{1 - x^2}}$	$y := \arctan(x / sqrt(1 - sqr(x)))$

$y = \arccos x = \frac{\pi}{2} - arctg\frac{x}{\sqrt{1-x^2}}$	$y := PI/2 - \arctan(x/sqrt(1-sqr(x)))$
$y = arcctgx = \frac{\pi}{2} - arctgx$	$y := PI/2 - \arctan(x)$
$y = scx = \frac{1}{\cos x}$ (секанс)	$y := 1/\cos(x)$
$y = csx = \frac{1}{\sin x}$ (косеканс)	$y := 1/\sin(x)$

Определение принадлежности точки указанной области

Для определения принадлежности точки указанной области необходимо: 1) определить уравнение границ области;

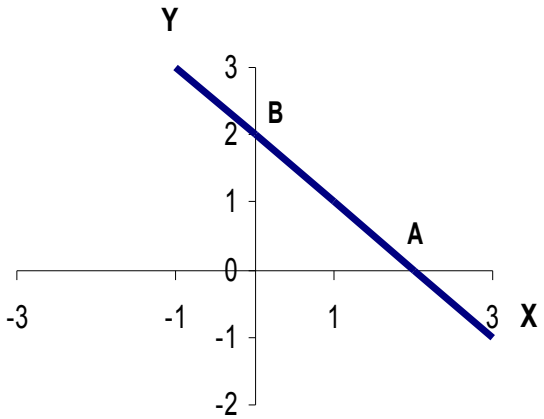
2) составить неравенства, определяющие заштрихованную область.

Пример1

Уравнение прямой:

Y=ax+b

Для определения коэффициентов a и b, рассмотрим точки пересечения прямой с осями X и Y.



1. Точка B (0;2)

Подставим координаты т.B в уравнение прямой и определим значение коэффициента b:

x=0, y=2

2=a·0+b

b=2

2. Точка A (2;0)

Подставим координаты т.A и значение коэффициента b=2 в уравнение прямой и определим значение коэффициента a:

$0=a\cdot 2+2$

$a=-1$

Следовательно, уравнение прямой будет иметь следующий вид:

$$y=-x+2.$$

Для всех точек, находящихся снизу от прямой, будет выполняться следующее неравенство, определяющее условие заштрихованной области:

$$y\leq -x+2.$$

Пример 2

Уравнение границ:

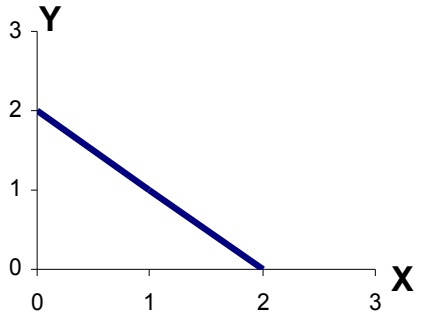
$y=-x+2;\ x=0;\ y=0$

Условия заштрихованной области:

$y\leq -x+2;$

$x\geq 0;$

$y\geq 0$

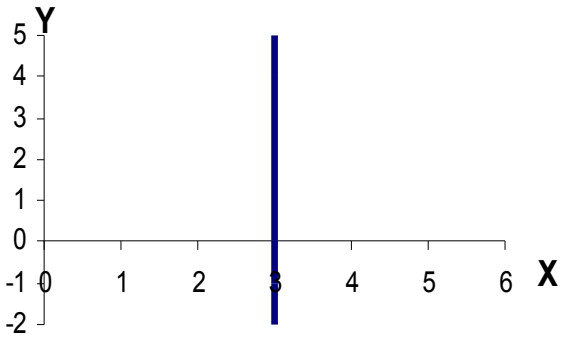


Пример 3

Уравнение границы:

$x=3$

Условие заштрихованной области: $x\geq 3$



Пример 4

Уравнение границ:

$x=5;\ x=10;\ y=-1;\ y=3$

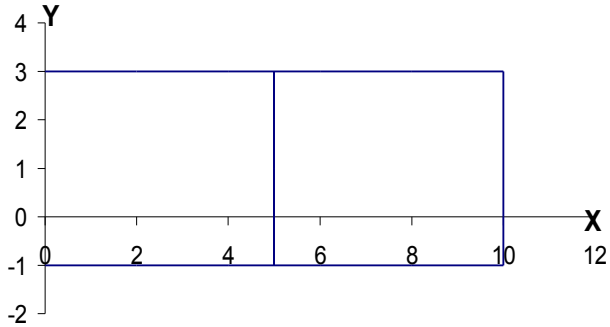
Условие заштрихованной области:

$x\geq 5;$

$x\leq 10;$

$y\geq -1;$

$y\leq 3$



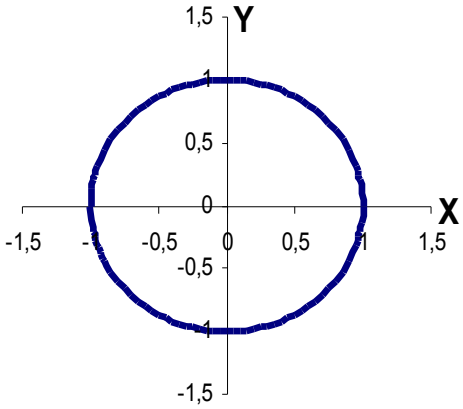
Пример 5

Уравнение границ:

$x^2 + y^2 = 1$

Условие заштрихованной области:

$x^2 + y^2 \le 1$



Пример 6

Уравнение границ:

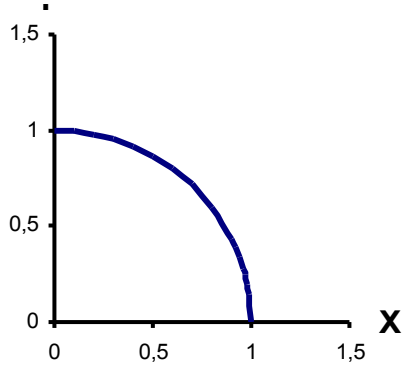
$x^2 + y^2 = 1;\ x = 0;\ y = 0$

Условие заштрихованной области:

$x^2 + y^2 \le 1;$

$x \ge 0;$

$y \ge 0$



ЗАДАНИЕ

Вариант 1

1. $z = 2^{-t} \cdot \sqrt{t + \sqrt[4]{|t|}}$ при $t = 4{,}741$

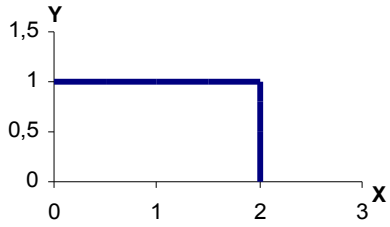
2. z*1* – соответствует модулю *z*;

z2– число, полученное при округлении *z*;

z3 - определяет четное ли значение z2;

z4– символ, стоящий под кодом 136.

3. (0,5;0,5)



Самостоятельная работа

1. Используя таблицу стадартных фунций, рассчитать любые значение переменных z5, z6, z7.

2. Изменить границы заштрихованной области и координаты точки. Определить принадлежность точки данной заштрихованной области.

3. Вывод результатов осуществить в новый файл результатов.

ЛАБОРАТОРНАЯ РАБОТА № 29

ТУРБО ПАСКАЛЬ. РАБОТА С МАССИВАМИ. ОПЕРАТОР ЦИКЛА FOR

ЗАДАНИЕ

Разработать программу по индивидуальному заданию. В программе использовать:

1. Описание массива любым способом (через раздел констант, через раздел типов или через раздел переменных).

2. Файловый ввод данных. При этом файл исходных данных должен иметь вид:

N

a1 a2 a3 ... ak

где N – число элементов массива

a1, ..., ak – значения элементов массива.

3. Считывание данных из файла исходных данных с помощью оператора цикла, например:

```
for i:=1 to N do read (f1, A[i]);
```

где i – переменная, которая считает элементы массива и организует цикл;

f1 – файловая переменная, связанная с файлом исходных данных;

A[i] – элементы исходного массива.

4. Преобразование элементов исходного массива A[i] в элементы нового массива B[i], например:

```
B[i]:=A[i]*5;
```

где каждый элемент нового массива B[i] больше соответствующего элемента исходного массива A[i] в 5 раз.

Замечание

Любые действия над элементами массива рекомендуется выполнять в рамках оператора цикла:

```
for i:=1 to N do begin ... end;
```

5. Файловый вывод данных.

Пример файла результатов:

Исходный массив:

1 2 3 4 5

Массив результатов 1:

25 10 15 20 25

Массив результатов 2:

0.1 0.2 0.3 0.4 0.5

Пример программы увеличение элементов исходного массива A в 5 раз с использованием файлового ввода данных.

Файл исходных данных GGG.TXT имеет вид:

5

1 2 3 4 5

```
PROGRAM massiv;{ Увеличение элементов массива в 5 раз }
VAR a,d:array[1..5]of integer; N,i:integer;  f1:text;
BEGIN
 assign (f1,'GGG.TXT');
reset(f1);
 readln (f1,N);     {Считывание из файла исходных данных количества
                    элементов массива}
 writeln;
 writeln('Исходный массив:');
 for i:=1 to N do
    begin
       read(f1,A[i]); {Считывание значений каждого
                          элемента исходного массива}
       write(A[i]:4); {Вывод на экран каждого элемента
                       исходного массива}
    end;
```

```
  writeln;
  writeln('Массив результатов 1:');
  for i:=1 to N do
     begin
        D[i]:=A[i]*5; {Вычисление значений каждого
                       элемента массива результатов}
        write(D[i]:4); {Вывод на экран каждого элемента
                        массива результатов}
     end;
END.
```

Самостоятельная работа

1. Найти массив результатов 3 самостоятельно любым способом.

2. Вывести все полученные массивы в новый файл результатов.

3. Расположить полученные массивы в файле в строчку, в столбик, используя таблицу.

ЛАБОРАТОРНАЯ РАБОТА № 30

ТУРБО ПАСКАЛЬ
РАБОТА С ДВУМЕРНЫМИ МАССИВАМИ

ЗАДАНИЕ

Точки A(x1, y1), B(x2, y2) и C(x3, y3) перемещаются, изменяя свои координаты. Известны их координаты в 8 моментах времени (см.карточку индивидуального задания).

1. Найти расстояние AB, AC, BC в каждый момент времени и вывести его на экран в виде таблицы.

Момент времени	AB	AC	BC
1			
2			

…			
7			
8			

Расстояние между точками A(x1, y1) и B(x2, y2) определяется по формуле:

$$AB = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \qquad (1)$$

С учетом того, что расстояния AB, AC и BC в данном случае будут представлять одномерный массив, то их можно определить:

```
AB[i]:=sqrt(sqr(x1[i,j]-x2[i,j])+sqr(y1[i,j]-y2[i,j]));
```

2. Ввод элементов исходного массива осуществлять из файла.

Считывание элементов из файла и вывод результатов на экран осуществлять оператором цикла.

Пример организации считывания данных из файла ttt.txt вида:

A11 A12 A13 A14

A21 A22 A23 A24

где Aij – значения элементов массива;

При этом первый столбец можно считать координатой x1, второй – y1, и т.д.

i – номер строки массива;

j – номер столбца массива.

С учетом выбранной формы файла исходных данных формула (1) запишется в виде:

```
AB[i]:=sqrt(sqr(A[i,1]-A[i,3])+sqr(A[i,2]-A[i,4]));
```

Фрагмент программы:

```
VAR A:array(1..2,1..3)of real;
   f1:text; i,j:integer;
BEGIN
   assign(f1,'ttt.txt'); reset(f1);
   for i:=1 to 2 do
```

```
      begin
        for j:=1 to 3 do read(f1,A[i,j]);
        readln(f1);
      end;
  .....
    close(f1);
  END.
```

3. После отладки программы организовать вывод результатов в файл.

Самостоятельная работа

1. В файл исходных данных добавить произвольные координаты точки D.

2. Рассчитать расстояния AD, BD, CD в каждый момент времени, вывести его в таблицу.

3. Создать новый файл результатов, в котором расположить таблицу результатов в горизонтальном виде.

ЛАБОРАТОРНАЯ РАБОТА №31

ТУРБО ПАСКАЛЬ
УСЛОВНЫЕ ОПЕРАТОРЫ

ЗАДАНИЕ

1. Составить блок-схему работы программы для выполнения следующего задания:

$$y = \begin{cases} 25 \ln 2x & \text{x} \le 2 \\ x^3 + 2x, & \text{если} \quad 2 < \text{x} < 8 \\ \sqrt{x} + x^2 & \text{в остальных случаях} \end{cases}$$

2. Создать программу расчета по предложенному заданию с организацией ввода исходных данных с клавиатуры и вывода результатов на экран.

3. Проверить работоспособность программы.
4. Отредактировать программу с организацией вывода результатов в файл. В файле результатов должны быть указаны исходные данные для расчета и комментарии к результатам расчета.
5. Оформить в тетради текст программы.
6. Записать в тетрадь имя файла результатов работы программы и его содержание.

Пример использования условного оператора

Дано значение ***x***, рассчитать значение величины ***w*** в соответствии с заданием

$$w = \begin{cases} 0 & x \le 0 \\ 2x & \text{если } 0 < x < 4 \\ x^2 & 4 \le x < 10 \\ x & \text{в остальных случаях} \end{cases}$$

С целью правильного оформления условного оператора необходимо построить блок-схему, после чего разработать программу расчета.

В приведенном ниже блоке программы предлагается один из вариантов использования условного оператора:

```
BEGIN
…
if x<=0  then w:=0
        else if (x>0) and (x<4) then w:=2*x
                                else if (x>=4) and (x<10) then w:=sqr(x)
                                                         else w:=x;
```

Самостоятельная работа

Создать программу для индивидуального задания:

Вывести первое число, если оно больше второго, и оба числа, если это не так.

ЛАБОРАТОРНАЯ РАБОТА № 32

ТУРБО ПАСКАЛЬ
ОПЕРАТОРЫ ЦИКЛА С ПОСТУСЛОВИЕМ И С ПРЕДУСЛОВИЕМ

Вариант 1

1. Составить блок-схему для определения поверхностной плотности трикотажного полотна в зависимости от выбранного вида пряжи. Полотно выполнено переплетением интерлок из хлопчатобумажной или шерстяной пряжи линейной плотности 16,5 текс. Алгоритм определения поверхностной плотности, Ps, г/м^2, приведен ниже.

$$l = \frac{\lambda \cdot \sqrt{T}}{31{,}62},$$

где $\lambda = 23$ для ххлопк

$\lambda = 24$ для шерсти

$$A = 0{,}13 \cdot l + 0{,}1\sqrt{T}$$

$$B = 0{,}35 \cdot l - 0{,}09\sqrt{T}$$

$$Пг = \frac{100}{A}$$

$$Пв = \frac{100}{B}$$

$$Ps = 10^{-4} \cdot Пг \cdot Пв \cdot l \cdot T$$

2.Записать условный оператор, реализующий вычисление по полученной блок-схеме.

3.Создать программу с организацией ввода исходных данных с клавиатуры и вывода результатов расчета на экран.

4.Проверить работоспособность программы.

5.Отладить программу с организацией расчета для пряжи линейной плотности в пределах от 16,5 до 29 текс с использованием оператора цикла с постусловием. Расчеты производить с шагом 1,5 текс.

6.Проверить работоспособность программы.

7.Отладить программу с организацией ввода исходных данных из файла и вывода результатов расчета в файл.

8. Выполнить расчеты по предложенному заданию.

9. Оформить отчет по работе в тетради с полным представлением окончательного варианта текста программы, указанием имени и содержания файлов исходных данных и результатов расчета.

Пример использования оператора с постусловием

Задание

Рассчитать запас материала на операции разгрузки, приемки и складирования. Расчет выполнятся по следующей формуле:

$$z = \frac{K \cdot R \cdot D}{P} \cdot 100,$$

где К - коэффициент запаса, К = 0,025

R - однодневный расход, $R = 800 м^2$

D - интервал поставок, D = 1- 17 дней, расчет производить с шагом 1 день.

P - число рабочих, занятых данной работой , P = 1

Фрагмент программы

```
…
  d1:=1;d2:=17;h:=1
  repeat
    z:= k*R*d1/P*100;
    Write('d=',d:4:2,'  ');
    Writeln('z=',z);
    d1:=d1+h
  until d1>d2;
…
```

Самостоятельная работа

1. Создать программу для данного задания, используя оператор цикла с предусловием.

2. Вывести результаты работы программы в виде таблицы в новый файл результатов.

ЛАБОРАТОРНАЯ РАБОТА № 33
ТУРБО ПАСКАЛЬ
ОПЕРАТОР ВЫБОРА CASE

Пример использования оператора CASE

Программа определения времени года рождения в зависимости от номера месяца рождения:

```
PROGRAM prim_4;
VAR x:integer;
BEGIN
  writeln('В какой месяц годы Вы родились?');
  write('Мой месяц рождения - ');
  readln(x);
  writeln('Время года Вашего рождения - ');
  case x of
    1,2,12:writeln(' зима');
    3,4,5:writeln(' весна');
    6,7,8:writeln(' лето');
    9,10,11:writeln(' осень');
  end;
END.
```

Отчет по работе программы

```
В какой месяц годы Вы родились?
Мой месяц рождения - 7
Время года Вашего рождения -  лето
```

ЗАДАНИЕ

Вариант 1

Определить наименование групп тканей по нумерации первых двух цифр артикулов льняных тканей. Исходные данные для расчета приведены в таблице.

Первые две цифры артикула	Наименование группы ткани
01, 02	Жаккардовые
03	Холсты и полотенца гладкие
06	Костюмно-платьевые ткани
10	Бортовые
11	Парусины
13	Равентухи
15, 16	Мешочные и мешки

Примечание: желательно переменную значения артикула описать типом «диапазон»

```
TYPE artikul=00..200;
VAR x: artikul;
```

Самостоятельная работа

1. Создать программу для определения курса обучения в зависимости от номера группы.

2. Создать программу для определения факультета в зависимости от номера группы.

2.5. Курсовая работа

Курсовая работа предусмотрена для закрепления знаний по основам информатики и навыков программирования в среде Турбо Паскаль.

Пояснительная записка к курсовой работе выполняется (набирается) на компьютере в текстовом редакторе WORD.

Курсовая работа содержит необходимые в соответствии с требованиями нормоконтроля разделы.

Структура курсовой работы

Введение (указываются цели и задачи курсовой работы)

1. Стандартный модуль Crt.

   1.1. …

   1.2. …

   …

2. Стандартный модуль Craph.

   2.1. …

   2.2. …

   …

3. Программирование на языке Turbo Pascal …

   3.1. Создание программ для расчета значения функции с заданным … диапазоном и шагом изменения аргумента

      3.1.1.Программа с использованием оператора цикла с … постусловием

      3.1.2.Программа с использованием оператора цикла с … предусловием

      3.1.3.Программа с использованием процедур и функций модуля Crt

   3.2. Создание программ для расчета статистических характеристик

      3.2.1.Программа для расчета статистических характеристик с файловым вводом данных

      3.2.2.Программа для расчета статистических характеристик с файловым выводом данных

      3.2.3.Программа для расчета статистических характеристик с использованием процедур и функций модуля Graph

   3.3.Создание программы с использованием условных операторов

   3.4. Создание программы с использованием оператора цикла FOR

   3.5. Создание программы для реализации процесса тестирования …

      3.5.1.Программа тестирования с вопросами, представленными в файле исходных данных

3.5.2.Программа тестирования с вопросами, представленными в программе

3.5.3.Программа тестирования с использованием процедур и функций модуля Crt

Заключение (делаются выводы по работе)

Список использованных источников

Приложения (приводятся распечатки всех программ)

Объем пояснительной записки к курсовой работе должен составлять 20-25 страниц печатного текста.

ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

№ недели	Дата	Раздел курсовой работы
1,2	1.09-10.09	Выдача задания к курсовой работе
3,4	11.09-23.09	1. Стандартный модуль Crt
5,6	25.09-7.10	2. Стандартный модуль Craph
		Контрольная точка 1
7,8	9.10-21.10	3.1. Создание программ для расчета значения функции с заданным диапазоном и шагом изменения аргумента
9,10	23.10-4.11	3.2 Создание программ для расчета статистических характеристик
		Контрольная точка 2
11,12	6.11-18.11	3.3 Создание программы с использованием условных операторов 3.4. Создание программы с использованием оператора цикла FOR
13,14,15	20.11-9.12	5. Создание программы для реализации процесса тестирования
		Контрольная точка 3
16	11.12-16.12	Оформление курсовой работы
17,18	18.12-30.12	Защита курсовой работы

2.6. Контроль знаний студентов

2.6.1. Перечень форм контроля

Промежуточный контроль знаний студентов осуществляется при подготовке к работе, выполнении и сдаче каждого задания лабораторной работы, а так же во время контрольных точек при выполнении курсовой работы.

В качестве заключительного контроля знаний студентов в 1 семестре служит зачет. На зачете оцениваются знания студентов по выполненным лабораторным работам. Во 2 семестре заключительным контролем служит экзамен, в 3 семестре – курсовая работа. К экзамену допускаются студенты при выполнении и защите всех лабораторных работ.

2.6.2. Оценка знаний студентов

Нормы оценки знаний предполагают учет индивидуальных особенностей студентов, дифференцированный подход к обучению, проверке знаний, умений.

В устных ответах студентов на экзамене и при защите курсовой работы учитываются: глубина знаний, полнота знаний и владение необходимыми умениями (в объеме полной программы); осознанность и самостоятельность применения знаний и способов учебной деятельности, логичность изложения материала, включая обобщения, выводы (в соответствии с заданным вопросом), соблюдение норм литературной речи.

2.6.3. Критерии оценки

Оценка знаний на экзамене и при защите курсовой работы производится по четырех балльной системе.

Оценка "пять" – материал усвоен в полном объеме; изложен логично; основные умения сформулированы и устойчивы; выводы и обобщения точны.

Оценка "четыре" – в усвоении материала незначительные пробелы, изложение недостаточно систематизированное; отдельные умения недостаточно устойчивы; в выводах и обобщениях допускаются некоторые неточности.

Оценка "три" – в усвоении материала имеются пробелы: материал изла-

гается несистематизированно; отдельные умения недостаточно сформулированы; выводы и обобщения аргументированы слабо; в них допускаются ошибки.

Оценка "два" – основное содержание материала не усвоено, выводов и обобщений нет.

2.7. Перечень экзаменационных вопросов

1) Понятие информации. Свойства информации. Место и роль понятия «информация» в курсе информатики.
2) Кодированная информация.
3) Понятие носителя информации. Формы представления и передачи информации.
4) Назначение компьютера.
5) История развития вычислительных средств.
6) Основные функциональные части компьютера.
7) Основные принципы работы компьютера.
8) Программа как последовательность действий компьютера. Понятие о машинном языке и языке Ассемблер.
9) Аппаратура компьютера.
10) Архитектура компьютера.
11) Центральные устройства компьютера.
12) Понятие алгоритма и алгоритмической системы.
13) Линейные, разветвленные и циклические алгоритмы. Вложенные и параллельные алгоритмы.
14) Компьютер как исполнитель алгоритмов.
15) Программа как изображение алгоритма в терминах команд, управляющих работой компьютера.
16) Коды, ассемблеры, языки высокого уровня. Трансляция и компоновка.
17) Программное обеспечение компьютера.
18) Системное и прикладное программное обеспечение.

19)Понятие об операционной системе.

20)Прикладное программное обеспечение.

21) Понятие языка высокого уровня. Синтаксис и семантика. Элементы и структуры данных, алфавит, имена, выражения, операции, операторы языка программирования Паскаль

22) Структура программ, аппарат подпрограмм, реализация логических структур языка программирования Паскаль.

23) Понятие программного продукта. Жизненный цикл программного обеспечения.

24)Задачи, решаемые с помощью баз данных.

25) Понятие телекоммуникации. Компьютерные сети как средство реализации практических потребностей.

26) Локальные сети и глобальные сети: принципы построения, архитектура, основные компоненты, их назначение и функции.

27)Всемирная компьютерная сеть ИНТЕРНЕТ. Ее возможности.

28) Модели решения функциональных и вычислительных задач.

29)Компьютерная графика

30)Системы геометрического моделирования.

31)Интегрированные автоматизированные системы

32) Угрозы безопасности информации и их классификация. Основные виды защищаемой информации. Проблемы ИБ в мировом сообществе.

33) Организационные меры, инженерно-технические и иные методы защиты информации, в том числе сведений, составляющих государственную тайну.

34) Защита информации в локальных компьютерных сетях, антивирусная защита.

2.8. Тесты для проверки остаточных знаний по Интернет - тестированию

ВАРИАНТ 1

1. Отдельные документы и отдельные массивы документов, документы и массивы документов в информационных системах (библиотеках, архивах, фондах, банков данных, других информационных системах) – это…

- ☐ информационные ресурсы
- ☐ информационный продукт
- ☐ база данных

2. Какой из способов подключения к Интернет обеспечивает наибольшие возможности для доступа к информационным ресурсам?

- ☐ постоянное соединение по оптоволоконному каналу
- ☐ удаленный доступ по коммутируемому телефонному каналу
- ☐ постоянное соединение по выделенному телефонному каналу
- ☐ терминальное соединение по коммутируемому телефонному каналу

3. Модем – это...

- ☐ почтовая программа
- ☐ сетевой протокол
- ☐ сервер Интернет
- ☐ техническое устройство

4. Электронная почта (e-mail) позволяет передавать...

- ☐ только сообщения
- ☐ только файлы
- ☐ сообщения и приложенные файлы
- ☐ видеоизображения

5. Какой протокол является базовым в Интернет?

- ☐ HTTP
- ☐ HTML
- ☐ TCP
- ☐ TCP/IP

6. Задан адрес электронной почты в сети Internet: user_name@int.glasnet.ru. Каково имя владельца электронного адреса?

- ☐ int.glasnet.ru
- ☐ user_name
- ☐ glasnet.ru
- ☐ ru

7. Браузеры (например, Microsoft Internet Explorer) являются...
   - ☐ серверами Интернет
   - ☐ антивирусными программами
   - ☐ трансляторами языка программирования
   - ☐ средством просмотра web-страниц
8. Web-страницы имеют формат (расширение)...
   - ☐ *.txt
   - ☐ *.htm
   - ☐ *.doc
   - ☐ *.exe
9. Группа компьютеров, связанных каналами передачи информации и находящихся в пределах территории, ограниченной небольшими размерами: комнаты, здания, предприятия, называется:
   - ☐ глобальной компьютерной сетью;
   - ☐ информационной системой с гиперсвязями;
   - ☐ локальной компьютерной сетью;
   - ☐ электронной почтой;
   - ☐ региональной компьютерной сетью?
10. Сетевой протокол- это:
    - ☐ набор соглашений о взаимодействиях в компьютерной сети;
    - ☐ последовательная запись событий, происходящих в компьютерной сети;
    - ☐ правила интерпретации данных, передаваемых по сети;
    - ☐ правила установления связи между двумя компьютерами в сети;
    - ☐ согласование различных процессов во времени.
11. Компьютер, подключенный к Интернет, обязательно имеет:

- ☐ IP-адрес;
- ☐ web-страницу;
- ☐ домашнюю web-страницу;
- ☐ доменное имя;
- ☐ URL-адрес.

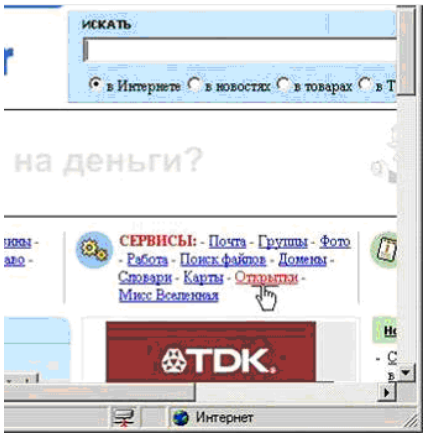
12. Телеконференция – это:
    - ☐ обмен письмами в глобальных сетях;
    - ☐ информационная система в гиперсвязях;
    - ☐ система обмена информацией между абонентами компьютерной сети;
    - ☐ служба приема и передачи файлов любого формата;
    - ☐ процесс создания, приема и передачи web-страниц.
13. HTML (HYPER TEXT MARKUP LANGUAGE) является:
    - ☐ язык разметки web-страниц;
    - ☐ системой программирования;
    - ☐ текстовым редактором;
    - ☐ системой управления базами данных;
    - ☐ экспертной системой.
14. News – является одной из рубрик телеконференций, выделяющей…
    - ☐ темы, связанные с компьютером
    - ☐ информация и новости
    - ☐ темы из области научных исследований
    - ☐ социальная тематика
    - ☐ дискуссии
15. Для чтения электронной почты предназначены следующие программы:
    - ☐ Outlook Express
    - ☐ The Bad
    - ☐ Windows XP
    - ☐ PhotoShop

16. Устройством для преобразования цифровых сигналов в аналоговую форму является…
    - ☐ концентратор
    - ☐ монитор
    - ☐ процессор
    - ☐ модем
17. Слайд – это…
    - ☐ совокупность объектов, расположенных на одной странице
    - ☐ отдельная страница презентации
    - ☐ объект презентации, содержащий графическую информацию
    - ☐ фоновый рисунок презентации
18. Программы архивации предназначены для архивирования…
    - ☐ только документов MS Office
    - ☐ музыкальных файлов
    - ☐ программ
    - ☐ видеофильмов
    - ☐ файлов в Интернет
    - ☐ любых файлов
19. Один килобайт содержит…
    - ☐ 1024 бита
    - ☐ 1024 байта
    - ☐ 1000 байт
    - ☐ 1000 символов
    - ☐ 1024 слов
20. Сканер – это…
    - ☐ команда
    - ☐ операционная системы
    - ☐ тест
    - ☐ устройство ввода данных
    - ☐ устройство поиска внеземных цивилизаций

21. Определите сервис Интернет, который не имеет собственного протокола и программы клиента

- ☐ www
- ☐ списки рассылки
- ☐ ICQ
- ☐ e-mail

22. Однократный щелчок левой кнопкой мыши в изображенном на рисунке окне Internet Explorer приведет к



- ☐ переходу к ресурсу cards.rambler.ru
- ☐ доставке электронной почты
- ☐ поиску документов по ключевым словам
- ☐ переходу к ресурсу www.rambler.ru
- ☐ открытию страницы новостей

23. Время отклика – это

- ☐ интервал, который проходит от момента передачи запроса на информацию с сервера, до момента начала ее получения
- ☐ интервал исполнения SQL запроса
- ☐ время на сервере в момент запроса
- ☐ время выполнения транзакции

24. Укажите варианты беспроводной связи…

- ☐ Ethernet
- ☐ Wi-FI
- ☐ IrDA
- ☐ FDDI

25. FTP-сервер – это…

- ☐ компьютер, на котором содержится информация для организации работы телеконференций
- ☐ корпоративный сервер

- ☐ компьютер, на котором содержатся файлы, предназначенные для открытого доступа
- ☐ компьютер, на котором содержатся файлы, предназначенные для администратора сети

26. Какие программы не относятся к антивирусным?
    - ☐ программы-фаги
    - ☐ программы сканирования
    - ☐ программы-ревизоры
    - ☐ прогаммы-детекторы
27. Заражению компьютерными вирусами могут подвергнуться...
    - ☐ графические файлы
    - ☐ программы и документы
    - ☐ звуковые файлы
    - ☐ видеофайлы
28. Компьютерный вирус – это…
    - ☐ неполадка в работе компьютера
    - ☐ программа, действующая без санкции пользователя
    - ☐ программа, стирающая все файлы на диске
    - ☐ программа, передаваемая по сети
29. Под утечкой информации понимается…
    - ☐ несанкционированный процесс переноса информации от источника к злоумышленнику
    - ☐ процесс раскрытия секретной информации
    - ☐ процесс уничтожения информации
    - ☐ непреднамеренная утрата носителя информации
30. Различают антивирусные программы…
    - ☐ ревизоры
    - ☐ репликаторы
    - ☐ детекторы или фаги
    - ☐ фильтры

31. База данных – это:
    - ☐ совокупность данных, организованных по определенным правилам;
    - ☐ совокупность программ для хранения и обработки больших массивов информации;
    - ☐ интерфейс, поддерживающий наполнение и манипулирование данными;
    - ☐ определенная совокупность информации.
32. Драйверы – это:
    - ☐ программы для ознакомления пользователя с принципами устройства компьютера;
    - ☐ программы для согласования работы внешних устройств и компьютера;
    - ☐ комплекс программ, обеспечивающий перевод программы, написанной на языке программирования в машинные коды;
    - ☐ системы автоматизированного проектирования;
    - ☐ технические устройства.
33. Наиболее точным определением понятия «переменная» в традиционных языках программирования является:
    - ☐ описание действий, которые должна выполнять программа;
    - ☐ именованная область памяти, в которой хранится некоторое значение;
    - ☐ любое законченное минимальное смысловое выражение на языке программирования;
    - ☐ служебное слово на языке программирования.
34. Оперативное запоминающее устройство относится к _____ памяти:
    - ☐ вспомогательной;
    - ☐ внутренней;
    - ☐ виртуальной;
    - ☐ дополнительной;

- ☐ внешней.

35. В чем состоит особенность поля «мемо»?
    - ☐ служит для ввода числовых данных;
    - ☐ служит для ввода действительных чисел;
    - ☐ данные хранятся не в поле, а в другом месте, а в поле хранится только указатель на то, где расположен текст;
    - ☐ имеет ограниченный размер;
    - ☐ имеет свойство автоматического наращивания.
36. Ключами поиска в системах управления базами данных (СУБД) называются:
    - ☐ диапазон записей файла БД, в котором осуществляется поиск;
    - ☐ логические выражения, определяющие условия поиска;
    - ☐ поля, по значению которых осуществляется поиск;
    - ☐ номера записей, удовлетворяющих условиям поиска;
    - ☐ номер первой по порядку записи, удовлетворяющей условиям поиска?
37. Аналогом элемента реляционной базы данных является …
    - ☐ файл;
    - ☐ папка;
    - ☐ вектор;
    - ☐ двумерная таблица.
38. СУБД – это …
    - ☐ свойства удаленной базы данных;
    - ☐ система удаления блокированных данных;
    - ☐ система управления большими данными;
    - ☐ система управления базами данных.
39. Вы открыли таблицу и решили изменить шрифт в одном из столбцов таблицы. Для этого Вы выделили нужный столбец, дали команду Формат ▸ Шрифт и выбрали другой шрифт. Что Вы увидите на экране?
    - ☐ изменение шрифта произошло в выделенном столбце таблицы;

☐ изменение шрифта произошло во всей таблице;

40. Вы открыли таблицу с информацией о сотрудниках, в которой есть поле «Дата рождения». Вам нужно найти запись в таблице, относящуюся к самому молодому сотруднику. Какой из способов является самым быстрым?

□ установить курсор ввода в любую ячейку столбца «Дата рождения» и щелкнуть кнопку Сортировка по возрастанию на панели инструментов;

□ установить курсор ввода в любую ячейку столбца «Дата рождения» и щелкнуть кнопку Сортировка по убыванию на панели инструментов;

☐ использовать команду Найти;

☐ применить фильтр;

41. Представлена таблица базы данных «Телефонный справочник». После проведения сортировки по полю «Фамилия» в порядке возрастания запись, содержащая номер телефона 384-15-15 переместится на …

	Фамилия	Имя	Отчество	Телефон
	Иванов	Николай	Васильевич	234-56-98
	Сидоров	Александр	Иванович	235-60-07
	Петров	Иван	Денисович	435-88-78
	Скворцов	Артем	Георгиевич	568-98-00
	Трофимов	Михаил	Валерьевич	384-15-15

☐ 4 строки вверх;

☐ 2 строки вверх;

☐ 3 строки вверх;

☐ 1 строку вверх;

☐ не переместится.

42. Представлена таблица базы данных «Литература». Запросу (Серия=«Для чайников») или (Количество страниц>299) в этой базе данных удовлетворяют …

	Автор	Серия	Название	Год	Кол_стр
	Уолш Р.	Для начинающих	Windows 95	1996	128
	Султанов И.	Для начинающих	Энциклопедия Delphi	1997	300
	Кирсанов Д.	Для чайников	Word 7.0	1996	236
	Харвей Г.	Для чайников	Excel 2000	2001	382
	Сигел Ч.	Изучи сам	Access 97	1998	352
	Визо М.	Компьютер для носорогов	Access 2.0	1994	255

☐ три записи;

☐ пять записей;

☐ одна запись;

☐ ни одной записи;

☐ четыре записи.

3. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

3.1. Основная литература

1. Информатика.Базовый курс. Под редакцией Симоновича С.В. – СПб: Питер, 2001.
2. Информатика**:** Учебник. – 3-е перераб. изд. /Под ред. проф. Н.В. Макаровой. – М.: Финансы и статистика, 2001. – 768 с.: ил.
3. Основы современных компьютерных технологий. Под редакцией А.Д.-Хомоненко. – СПб: Корона-принт, 1998.
4. Тюрин Ю.Н., Макаров А.А. Статистический анализ данных на компьютере. – М.:ИНФРА. 1998.
5. Герасименко В.А., Малюк А.А. Основы защиты информации. – М., 1997.

3.2. Дополнительная литература

1. Громов Г.Р. Национальные информационные ресурсы: проблемы промышленной эксплуатации. - М.: Наука. 1984, 1985.- 237
2. Фигурнов В.Э. IBM PC для пользователя. Краткий курс. М.: Финансы и статистика. 1997.
3. Колесниченко С., Шишигин И. Аппаратные средства PC. BHV 1999.
4. Мэтьюз М.. Microsoft Windows 98, Спутник пользователя. Microsoft Press 2000.
5. MS Word 97 (2000). Шаг за шагом. Ecom 1999 (2000).
6. MS Excel 97 (2000). Шаг за шагом. Ecom 1999 (2000).
7. MS Access 97 (2000). Шаг за шагом. Ecom 1999 (2000).
8. Дубнов П.Ю. Access 2000. Проектирование баз данных. Ecom 2000.
9. Кудрявцев Е.М.. MathCad 8. ДМК М. 2000
10. Лазарев Л. MatLab 5.x. Библиотека студента. BHV 2000.

11. Шураков В.В., Дайитбеков Д.М., Мизрохи С.В., Ясеновский С.В. Автоматизированное рабочее место для статистической обработки данных. М., Финансы и статистика, 1990.
12. Бобровский С. Программирование на языке QBasic для школьников и студентов.
13. Кулагин Н.Б. Программирование в Turbo Pascal 7.0 и Delphy. BHV 2000
14. Паппас К., Мюррей У. Программирование на С и С++. Библиотека студента. BHV 2000.
15. Б. Керниган, Д. Ритчи. Язык программирования Си (пер. с англ.). — М.: Финансы и статистика, 1992.
16. Ю. Тихомиров. Visual C++ 6. — Киев: BHV, 1998
17. Турбо Паскаль. — Киев: BHV, 1996.
18. А. Епанешников, В. Епанешников. DELPHI 4. Среда разработки: Учебное пособие. — М.: Диалог МИФИ, 1999.
19. Искусственный интеллект. Справочник в 3-х томах под. ред. Попова Э.В. Т 1. Системы общения и экспертные системы. М. Радио и связь, 1990.
20. Уотерман Д. Руководство по экспертным системам. М.: Мир. 1989.
21. Виноградов С.М. и др. Информационное общество: Информационные войны. Информационное управление. Информационная безопасность. Изд. СПбУ, 1999.
22. Копылов В.А. Информационное право. М., “Юристъ”, 1997.
23. Федеральный закон Российской Федерации “Об информации, информатизации и защите информации” от 20.02.1995 № 24-ФЗ.
24. Закон Российской Федерации “О государственной тайне” от 21.07.1993 № 5485-1 с изменениями и дополнениями от 06.10.1997.

3.3. Методическое обеспечение дисциплины

1. Работа в операционных средах MS DOS и Windows.Учебно-методическое пособие. – Благовещенск, АмГУ, 1998.

2. Чалкина А.Н., Масленко Т.В. Практикум по решению задач на языке Turbo Pascal 7.0 по курсу “Информатика”. – Благовещенск: Изд-во АмГУ, 1999.

3. Абакумова И.В., Тибенко Т.А., Сухова Т.Н. Обработка данных средствами Excel. Учебно-методическое пособие. Амурский гос.ун-т, Благовещенск, 2006.

3.4 Технические средства обеспечения дисциплины

1. Компьютер.

4. УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА ДИСЦИПЛИНЫ

4.1 Учебно-методическая карта дисциплины для специальности 260704, 260901, 260902

1 семестр

№ недели	№ темы лекции	Вопросы, рассматриваемые в лекции	№ лабораторной работы	Самостоятельная работа студентов		
				содержание	часы	форма контроля
1	2	3	5	6	7	8
1, 2	1	Понятие информации	1, 2	1. Работа в операционной среде Windows . 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
3, 4, 5	2	Принцип работы компьютера	3, 4, 5	1. Работа в операционной среде Windows . 2. Выполнение индивидуального практического задания.	4	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
6, 7	3	Аппаратура компьютера. Технические средства реализации информационных процессов	6, 7	1. Работа в операционной среде Windows . 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
8, 9,10	4	Алгоритмы и алгоритмизация. Визуализация алгоритмов	9, 10	1. Работа в операционной среде Windows . 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
11, 12	5	Программирование	11, 12	1. Работа с текстовым и графическим редактором. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
13,14,15	6	Программное обеспечение	13	1. Работа с текстовым процессором. 2. Выполнение индивидуального практического задания.	4	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
16, 17, 18	7	Обзор языков высокого уровня	21, 22	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания.	4	Защита лабораторной работы с предоставлением отчета и индивидуального задания.

4.2 Учебно-методическая карта дисциплины для специальности 260704, 260901, 260902

2 семестр

№ недели	№ темы лекции	Вопросы, рассматриваемые в лекции	№ лабораторной работы	Самостоятельная работа студентов		
				содержание	часы	форма контроля
1	2	3	5	6	7	8
1, 2, 3	7	Обзор языков высокого уровня	23, 24	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания.	4	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
4, 5	8	Технология программирования	25, 26	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
6, 7	9	Базы данных	18	1. Работа в СУБД Access. 2. Выполнение индивидуального практического задания.	4	Защита лабораторной работы с предоставлением отчета и индивидуального задания .
8, 9	10	Телекоммуникации	19	1. Работа в Интернете. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания .
10, 11	11	Модели решения функциональных и вычислительных задач	20	1. Работа в Интернете. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
12, 13	12	Компьютерная графика и системы геометрического моделирования	14	1. Работа с графическим редактором. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
14, 15	13	Интегрированные автоматизированные системы	14	1. Работа с графическим редактором. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
16, 17	14	Основы защиты информации	8	1. Работа в операционной среде Windows . 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.

4.3 Учебно-методическая карта дисциплины для специальности 260704, 260901, 260902

3 семестр

№ недели	№ темы лекции	Вопросы, рассматриваемые в лекции	№ лабораторной работы	Самостоятельная работа студентов		
				содержание	часы	форма контроля
1	2	3	5	6	7	8
1, 2			27	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
3, 4			28	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
5, 6			29	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
7, 8			30	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
9, 10			31	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
11, 12			32	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
13			33	1. Работа в интегрированной среде Турбо Паскаль. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
14			15	1. Работа со сканером и принтером. 2. Выполнение индивидуального практического задания	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
15, 16			16	1. Работа в Power Point. 2. Выполнение индивидуального практического задания.	2	Защита лабораторной работы с предоставлением отчета и индивидуального задания.
17, 18			17	1. Работа с электронной таблицей Excel. 2. Выполнение индивидуального практического задания.	3	Защита лабораторной работы с предоставлением отчета и индивидуального задания.

СОДЕРЖАНИЕ

1. Цели и задачи дисциплины, ее место в учебном процессе 3
2. Содержание дисциплины 5
2.1 Требование стандарта 5
2.1.1. Требование стандарта по направлению 656000 «Технология и проектирование текстильных изделий» 5
2.1.2. Требование стандарта по направлению 656100 «Технология и конструирование изделий легкой промышленности» 5
2.2 Наименование тем, объем (в часах) лекционных, лабораторных занятий и самостоятельной работы 6
2.3 План-конспект лекций 7
2.4 Лабораторные занятия. Самостоятельная работа студентов 294
Лабораторная работа № 1. Первоначальные сведения о среде WINDOWS 294
Лабораторная работа № 2. Работа с окнами WINDOWS 301
Лабораторная работа № 3. Клавиатура 304
Лабораторная работа № 4. Работа с файлами и каталогами в среде WINDOWS 307
Лабораторная работа № 5. Удаление, восстановление, поиск и сортировка записей в WINDOWS 313
Лабораторная работа № 6. Настройка свойств мыши и оформление рабочего стола 317
Лабораторная работа № 7. Проверка и дефрагментация диска 320
Лабораторная работа № 8. Резервное копирование данных и проверка диска 323
Лабораторная работа № 9. Программа FAR Manager 325
Лабораторная работа № 10. Программа Блокнот 328
Лабораторная работа № 11. Текстовый редактор WordPad 332
Лабораторная работа № 12. Графический редактор Paint 334
Лабораторная работа № 13. Текстовый процессор MS WORD 338

Лабораторная работа № 14. Технологии обработки графической информации 342
Лабораторная работа № 15. Технологии распознования информации. Работа со сканером и принтером 344
Лабораторная работа № 16. Создание и настройка Power Point 348
Лабораторная работа № 17. Создание таблиц и выполнение расчетов в программе MS Excel 351
Лабораторная работа № 18. Работа с базой данных СУБД ACCESS 356
Лабораторная работа № 19. Навигация и поиск информации в Интернете 377
Лабораторная работа № 20. Работа с электронной почтой и в телеконференциях 385
Лабораторная работа № 21. Турбо Паскаль. Экран интегрированной среды 403
Лабораторная работа № 22. Турбо Паскаль. Структура программы. Компиляция программы 410
Лабораторная работа № 23. Турбо Паскаль. Операторный ввод-вывод данных 413
Лабораторная работа № 24. Турбо Паскаль. Файловый ввод данных 415
Лабораторная работа № 25. Турбо Паскаль. Вывод результатов в различных вариантах 417
Лабораторная работа № 26. Турбо Паскаль. Файловый вывод данных 419
Лабораторная работа № 27. Турбо Паскаль. Работа с символьными и строковыми данными 420
Лабораторная работа № 28. Турбо Паскаль. Стандартные функции 422
Лабораторная работа № 29. Турбо Паскаль. Работа с массивами.

Оператор цикла FOR 427
Лабораторная работа № 30. Турбо Паскаль. Работа с двумерными массивами 429
Лабораторная работа № 31. Турбо Паскаль. Условные операторы 431
Лабораторная работа № 32. Турбо Паскаль. Операторы цикла с постусловием и с предусловием 433
Лабораторная работа № 33. Турбо Паскаль. Оператор выбора CASE 435
2.5. Курсовая работа 436
2.6. Контроль знаний студентов 438
2.6.1. Перечень форм контроля 438
2.6.2. Оценка знаний студентов 439
2.6.3. Критерии оценки 439
2.7. Перечень экзаменационных вопросов 440
2.8. Тесты для проверки остаточных знаний по Интернет-тестированию 441
3. Учебно-методические материалы по дисциплине. Рекомендуемая литература 451
3.1. Основная литература 451
3.2. Дополнительная литература 451
3.3 Методическое обеспечение дисциплины 452
3.4 Технические средства обеспечения дисциплины 453
4. Учебно-методическая карта дисциплины 454
4.1 Учебно-методическая карта дисциплины для специальностей 260704, 260901, 260902 1 семестр 454
4.2 Учебно-методическая карта дисциплины для специальностей 260704, 260901, 260902 2 семестр 455
4.3 Учебно-методическая карта дисциплины для специальностей 260704, 260901, 260902 3семестр 456

Ирина Валентиновна Абакумова,

доцент кафедры конструирования и технологии одежды АмГУ

Информатика
Учебно-методический комплекс по дисциплине для специальностей
260704 – «Технология текстильных изделий»,
260901 – «Технология швейных изделий»,
260902 – «Конструирование швейных изделий»,