

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»
Кафедра «Организация технологических процессов»**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ
ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ**

Б1.В.ДВ.09.01 Информатика

Направление подготовки 38.03.21 Менеджмент

Профиль образовательной программы Управленческий финансовый учет

Форма обучения очная

СОДЕРЖАНИЕ

1. Конспект лекций	3
1.1 Лекция № 1 Введение, основные понятия информатики	3
1.2 Лекция № 2 Понятие информации	7
1.3. Лекция №3 Технические средства реализации информационных процессов	11
1.4. Лекция №4 . Системное программное обеспечение.....	17
1.5. Лекция №5 Основные понятия и принципы моделирования	22
1.6. Лекция №6,7,8 Алгоритмизация вычислительных процессов	25
1.7. Лекция №9 Программные средства реализации информационных процессов.	33
1.8. Лекция №10,11 Программирование на алгоритмическом языке высокого уровня.....	40
1.9. Лекция №12,13Массивы. Обработка одномерных массивов	..44
1.10 Лекция №14 . Сервисное программное обеспечение	47
1.11 Лекция №15 . Прикладное программное обеспечение	50
1.12. Лекция №16,17 Компьютерные сети	52
1.13. Лекция №18 Защита информации	56
2. Методические указания по выполнению лабораторных работ	62
2.1. Лабораторная работа № ЛР-1 Понятие информации	62
2.2. Лабораторная работа № ЛР-2 Системное ПО	63
2.3. Лабораторная работа № ЛР-3,4 Системное ПО	69
2.4. Лабораторная работа № ЛР-5,6,7,8,9 Алгоритмизация вычислительных процессов с предусловием).	78
2.5. Лабораторная работа № ЛР-10,11,12 Программирование на алгоритмическом языке высокого уровня	80
2.6 Лабораторная работа 13, 14 (ЛР-13, ЛР-14) Массивы. Обработка одномерных массивов	
2.7. Лабораторная работа № ЛР-14 Сервисное ПО ...	...81
2.8. Лабораторная работа № ЛР-15Прикладное ПО	86
3. Методические указания по проведению практических занятий	98

1. КОНСПЕКТ ЛЕКЦИЙ

1. 1 Лекция №1 (2 часа).

Тема: «Введение, основные понятия информатики»

1.1.1 Вопросы лекции:

1. Этапы информатизации общества
2. Цель и задачи информатики. Основные понятия
3. Структура информатики
4. Социальные, правовые и этические аспекты информатики

1.1.2

1. Этапы информатизации общества

Бурное развитие компьютерной техники и ИТ послужило толчком к развитию общества, построенного на использовании различной информации и получившего название информационного общества. В информационном обществе изменяется не только производство, но и весь уклад жизни, система ценностей, возрастает значимость культурного досуга по отношению к материальным ценностям. По сравнению с индустриальным обществом, где все направлено на производство и потребление товаров, в информационном обществе производятся интеллект и знания. Материальной и технологической базой информационного общества становятся компьютерная техника и компьютерные сети, ИТ, телекоммуникационные связи.

Информационное общество – общество, в котором большинство работающих занято производством, хранением, переработкой и реализацией информации, особенно высшей ее формы – знаний.

Ближе всех на пути к информационному обществу стоят страны с развитой информационной индустрией (США, Япония, Англия, страны Западной Европы).

Для перехода от индустриального общества к информационному должна была возникнуть ситуация информационного кризиса. И она возникла с тем, что в 20 веке лавинообразный поток информации, хлынувший на человека, сделал практически невозможной его ориентацию в этом объеме. Возникло большое количество избыточной, лишней информации. Началом перехода к информационному обществу стало внедрение современных средств обработки и передачи информации в различных сферах деятельности человека. Этот процесс называется информатизация.

Информатизация общества – это процесс, при котором создаются условия, удовлетворяющие потребностям любого человека в получении необходимой информации (по закону РФ «Об информации, информатизации и защите информации» от 25 января 1995 года).

Цель информатизации – улучшение качества жизни людей за счет увеличения производительности и облегчения условий их труда.

2 Цель и задачи информатики. Основные понятия

Термин «информатика» прочно и надежно вошел в нашу жизнь. Он был заимствован из французского языка и обозначал название области, связанной с автоматизированной обработкой информации с помощью ЭВМ. Именно развитие компьютерной техники способствовало выделению информатики в самостоятельную область человеческой деятельности.

Следует отметить, что определений информатики в современной литературе множество. Это происходит оттого, что данная область знаний относительно новая и соответствующий понятийный аппарат не совсем устоялся. Анализ определений позволил выделить их существенную часть и сформулировать то определение, которое приведено ниже.

Информатика - область человеческой деятельности, связанная с процессами преобразования информации с помощью компьютеров и других средств вычислительной техники.

3 Структура информатики

С информатикой часто связывают одно из следующих понятий: это либо **отрасль производства**, либо **фундаментальная наука**, либо **прикладная дисциплина**, либо совокупность определенных средств, используемых для преобразования информации.

В состав **технических средств** входят компьютеры и связанные с ними периферийные устройства (мониторы, клавиатуры, принтеры и плоттеры, модемы и т.д.), линии связи, средства оргтехники и т.п., т.е. те материальные ресурсы, которые обеспечивают преобразование информации, причем главенствующую роль в этом списке играет компьютер.



Рис 1.1 Структура информатики

К **программным средствам** (продуктам) относятся операционные системы, интегрированные оболочки, системы программирования и проектирования программных продуктов, различные прикладные пакеты, такие, как текстовые и графические редакторы, бухгалтерские и издательские системы и т.д. **Математические методы, модели и алгоритмы** являются тем базисом, который положен в основу проектирования и изготовления любого программного или технического средства в силу их исключительной сложности и, как следствие, невозможности умозрительного подхода к созданию.

Перечисленные выше три ресурсных компонента информатики играют разную роль в процессе информатизации общества. Так, совокупность программных и технических средств, имеющихся в том или ином обществе, и позволяет сделать его информационным, когда каждый член общества имеет возможность получить практически любую (исключая, естественно, секретную) интересующую его информацию (такие потребители информации называются конечными пользователями). В то же время, сложность технических и программных систем заставляет использовать имеющиеся технические и программные продукты, а также нужные методы, модели и алгоритмы для проектирования и производства новых и совершенствования старых технических и программных систем. В этом случае можно сказать, что средства преобразования информации используются для производства себе подобных. Тогда их пользователем является специалист в области информатики, а не конечный пользователь.

Разработкой абстрактных методов, моделей и алгоритмов, а также связанных с ними математических теорий занимается **фундаментальная наука**. Ее прерогативой является исследование процессов преобразования информации и на основе этих исследований разработка соответствующих теорий, моделей, методов и алгоритмов, которые затем применяются на практике.

Практическое использование результатов исследований информатики как фундаментальной науки воплощается в информатике - **отрасли производства**. В самом деле, широко известны западные фирмы по производству программных продуктов, такие как Microsoft, Lotus, Borland, и технических средств - IBM, Apple, Intel, Hewlett Packard и другие. Помимо производства самих технических и программных средств разрабатываются также и технологии преобразования информации.

Подготовкой специалистов в области преобразования информации занимается информатика как **прикладная дисциплина**. Она изучает закономерности протекания информационных процессов в конкретных областях и методологии разработки конкретных информационных систем и технологий.

Таким образом, главная функция информатики состоит в разработке методов и средств преобразования информации с использованием компьютера, а также в применении их при организации технологического процесса преобразования информации.

В рамках **прикладной дисциплины** информатики изучаются следующие вопросы:

- понятие информации, ее свойства, измерение информации, использование в управлении;
- способы кодирования информации;
- понятие и составные части информационных процессов;
- организация технических устройств преобразования информации, в частности компьютера;
- структура и методология проектирования программного обеспечения.

Задачи информатики:

- Разработка и производство современных средств вычислительной техники;
- Проектирование и внедрение прогрессивных технологий обработки информации, и как результат этого – возможность дальнейшей информатизации общества и повышения уровня его информационной культуры.

4 Социальные, правовые и этические аспекты информатики

Социальные аспекты информатики

Термин “социальные аспекты” применительно к большей части наук. Однако, информатика – не только наука.

Мало какие факторы так влияют на социальную сферу обществ как информатизация.

Информатизация общества – процесс проникновения информационных технологий во все сферы жизни и деятельности общества. Она сильнее всего влияет на структуру экономики ведущих в экономическом отношении стран. В числе их лидирующих отраслей промышленности традиционные добывающие и обрабатывающие отрасли оттеснены максимально наукоемкими производствами электроники, средств связи и вычислительной техники. В этих странах постоянно растут капиталовложения в научные исследования, включая фундаментальные науки. Темпы развития сферы высоких технологий и уровень прибылей в ней превышают в 5-10 раз темпы развития традиционных отраслей производства. Такая политика имеет и социальные последствия – увеличение потребности в высокообразованных специалистах и связанный с этим прогресс системы высшего образования. Информатизация меняет и облик традиционных отраслей промышленности и сельского хозяйства. Промышленные роботы, управляемые ЭВМ, станки с ЧПУ стали обычным оборудованием. Новейшие технологии в сельскохозяйственном производстве не только увеличивают производительность труда, но и облегчают его, вовлекают более образованных людей.

Казалось бы, компьютеризация и информационные технологии несут в мир одну лишь благодать, но социальная сфера столь сложна, что последствия любого, даже гораздо менее глобального процесса, редко бывают однозначными. Рассмотрим, например, такие социальные последствия информатизации как рост производительности

труда, интенсификацию труда, изменение условий труда. Все это, с одной стороны, улучшает условия жизни многих людей, повышает степень материального и интеллектуального комфорта, стимулирует рост числа высокообразованных людей, а с другой – является источником повышенной социальной напряженности. Например, появление на производстве промышленных роботов ведет к полному изменению технологии, которая перестает быть ориентированной на человека. Тем самым меняется номенклатура профессий. Значительная часть людей вынуждена менять либо специальность, либо место работы – рост миграции населения характерен для большинства развитых стран. Государство и частные фирмы поддерживают систему повышения квалификации и переподготовки, но не все люди справляются с сопутствующим стрессом. Одним словом, жизнь в “информационном обществе” легче, по-видимому, не становится, а вот то, что она значительно меняется – несомненно

Правовые аспекты информатики

Деятельность программистов и других специалистов, работающих в сфере информатики, все чаще выступает в качестве объекта правового регулирования. Некоторые действия при этом могут быть квалифицированы как правонарушения (преступления).

Правовое сознание в целом, а в области информатики особенно, в нашем обществе находится на низком уровне. Все ли знают ответы на следующие вопросы:

- можно ли, не копируя купленную программу, предоставить возможность пользоваться ею другому лицу;
- кому принадлежит авторское право на программу, созданную студентом в ходе выполнения дипломной работы;
- можно ли скопировать купленную программу для себя самого, чтобы иметь резервную копию;
- можно ли декомпилировать программу, чтобы разобраться в ее деталях или исправить ошибки;
- в чем состоит разница между авторским и имущественным правом.

Вопросов, подобных этим, возникает множество, но остановимся на правовом регулировании в области информатики в России. К 1992 году был принят Закон Российской Федерации “О ПРАВОВОЙ ОХРАНЕ ПРОГРАММ ДЛЯ ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН И БАЗ ДАННЫХ”, содержащий обширный план приведения российского законодательства в сфере информатики в соответствие с мировой практикой. Действие этого Закона распространяется на отношения, связанные с созданием и использованием программ для ЭВМ и баз данных. Также предусматривалось внести изменения и дополнения в Гражданский кодекс РФ, в Уголовный кодекс РФ, другие законодательные акты, связанные с вопросами правовой охраны программ для электронных вычислительных машин и баз данных, привести решения Правительства РФ в соответствие с Законом, обеспечить пересмотр и отмену государственными ведомствами и другими организациями РФ их нормативных актов, противоречащих указанному Закону, обеспечить принятие нормативных актов в соответствии с указанным Законом и т.д. Главное содержание данного Закона – юридическое определение понятий, связанных с авторством и распространением компьютерных программ и баз данных, таких как Авторство, Адаптация, База данных, Воспроизведение, Декомпилирование. Использование, Модификация и т.д., а также установление прав, возникающих при создании программ и баз данных – авторских, имущественных, на передачу, защиту, регистрацию, неприкосновенность и т.д.

Этические аспекты информатики

Этика – система норм нравственного поведения человека.

Далеко не все правила, регламентирующие деятельность в сфере информатики, можно свести в правовым нормам. Очень многое определяется соблюдением неписаных правил поведения для тех, кто причастен к миру компьютеров. Впрочем, в этом

отношении информатика ничуть не отличается от любой другой сферы деятельности человека в обществе.

Морально-этические нормы в среде информатиков отличаются от этики повседневной жизни несколько большей открытостью, альтруизмом. Большинство нынешних специалистов-информатиков сформировались и приобрели свои знания и квалификацию благодаря бескорыстным консультациям и содействию других специалистов. Очевидно, поэтому они готовы оказать бескорыстную помощь, дать совет или консультацию, предоставить компьютер для выполнения каких-либо манипуляций с дискетами и т.д. Ярким примером особой психологической атмосферы в среде информатиков является расширяющееся международное движение программистов, предоставляющих созданные ими программные средства для свободного распространения.

1.2. Лекция № 2 (2 часа)

Тема: Понятие информации

1.1.2. Вопросы лекции:

1. Место и роль понятия «информация» в курсе информатики. Методы получения информации

2. Виды, свойства и формы представления информации

3. Классификация информации

4. Кодирование информации. Единицы измерения информации

2.1.2 Краткое содержание вопросов

1. Место и роль понятия «информация» в курсе информатики. Методы получения информации

В любой науке основные понятия определить достаточно сложно. Так и понятие «информация» невозможно определить через другие, более «простые» понятия. В каждой науке оно связано с различными системами понятий. Например, в определении понятия «информация» необходимо отразить, что она:

1. существует в неживой природе;
2. существует в биологических системах;
3. не всегда может быть выражена словами;
4. возникает в процессе общения;
5. хранится, обрабатывается, передается, и т.д.

В зависимости от области знания существуют различные подходы к определению понятия «информация».

Впервые как научное понятие термин «информация» стал применяться в теории журналистики в 30-х годах XX века, хотя в исследованиях по библиотечному делу он появился еще раньше. Под информацией понимались различные сведения, сообщения. Что соответствует переводу с латинского языка *informatio* – сведение, разъяснение, ознакомление.

В физике понятие информация рассматривается как антиэнтропия или энтропия с обратным знаком. Поскольку мерой беспорядка термодинамической системы является энтропия системы, то информация (антиэнтропия) является мерой упорядоченности и сложности системы. По мере увеличения сложности системы величина энтропии уменьшается, и величина информации увеличивается. Процесс увеличения информации характерен для открытых, обменивающихся веществом и энергией с окружающей средой, саморазвивающихся систем живой природы (белковых молекул, организмов, популяций животных и т.д.).

Информация - сведения об объектах и явлениях окружающей среды их параметрах, свойствах и состоянии, которые уменьшают степень неопределенности, имеющуюся о них в реальности.

Наряду с информацией в информатике часто употребляется понятие «данные». Данные обычно рассматриваются как признаки или записанные наблюдения, которые не используются, а только хранятся. В том случае, когда появляется возможность использовать данные для уменьшения степени неопределенности чего-либо, они превращаются в информацию.

Пример 1. Напишите на листе десять номеров телефонов в виде последовательности десяти чисел и покажите их вашему другу. Он воспримет эти цифры как данные, так как они не предоставляют ему никаких сведений.

Затем против каждого номера укажите название фирмы и род деятельности. Для вашего друга непонятные цифры обретут определенность и превратятся из данных в информацию, которую он в дальнейшем мог бы использовать.

Выделяют три фазы существования информации:

1. Ассимилированная информация - представление сообщений в сознании человека, накопленное на систему его понятий и оценок.
2. Документированная информация - сведения, зафиксированные в знаковой форме на каком-то физическом носителе.
3. Передаваемая информация - сведения, рассматриваемые в момент передачи информации от источника к приемнику.

При работе с информацией всегда имеется ее источник и потребитель (получатель). Пути и процессы, обеспечивающие передачу сообщений от источника информации к ее потребителю, называются *информационными коммуникациями*.

Содержание информационного процесса: сбор, передача, обработка и накопление информации

Сбор информации – это процесс получения информации из внешнего мира и приведение ее к стандарту для данной информационной системы. Обмен информацией между воспринимающей ее системой и окружающей средой осуществляется посредством сигналов. Сигнал – средство передачи информации в пространстве и времени. В качестве носителя сигнала могут выступать звук, свет, электрический ток, магнитное поле и т.д. Сбор информации, как правило, сопровождается ее регистрацией, т.е. фиксацией информации на материальном носителе (документе или машинном носителе).

Передача информации осуществляется различными способами: с помощью курьера, пересылка по почте, доставка транспортными средствами, дистанционная передача по каналам связи. Для осуществления последней необходимы специальные технические средства. Дистанционно может передаваться как первичная информация с мест ее возникновения, так и результатная информация в обратном направлении. Поступление информации по каналам связи осуществляется двумя способами: на машинном носителе и непосредственно в компьютер при помощи специальных программных и аппаратных средств.

Преобразование (обработка) информации – внесение изменений в набор данных, вычисления, информационный поиск, сортировка, построение графиков и т.п.

В современных развитых информационных системах машинная *обработка информации* предполагает последовательно-параллельное во времени решение вычислительных задач. Это возможно при наличии определенной организации вычислительного процесса. Вычислительная задача по мере необходимости обращается с запросами в вычислительную систему. Организация процесса предполагает определение последовательности решения задач и реализацию вычислений. Последовательность решения задается, исходя из информационной взаимосвязи, когда результаты решения одной задачи используются как исходные данные для решения другой.

2. Виды, свойства и формы представления информации

Все многообразие окружающей нас информации можно сгруппировать по различным признакам.

По признаку «*область возникновения*» информация делится на:

- элементарную – отражает процессы и явления неодушевленной природы;
- биологическую – отражает процессы растительного и животного мира;
- социальную – отражает процессы человеческого общества.

По *способу передачи и восприятия* различают информацию:

- визуальную – передается видимыми образами и символами;
- аудиальную – передается звуками;
- тактильную – передается ощущениями;
- органо-лептическую – передается запахами и вкусом;
- машинную – выдаваемую и воспринимаемую средствами вычислительной

техники.

По *форме представления* различают информацию:

- текстовую;
- числовую;
- графическую;
- музыкальную;
- комбинированную.

Информацию, создаваемую и используемую человеком, по *общественному назначению* делят на виды:

- массовая;
- специальная;
- личная.

В информатике рассматривают две формы представления информации:

1) аналоговую (непрерывную), изменяющуюся по амплитуде и по времени непрерывно, например температура тела.

2) дискретную (прерывистую), принимающую конечное число значений, например времена года.

Информация имеет определенные свойства:

1. Полнота информации отражает ее достаточность для понимания и для принятия управленческого решения.

2. Достоверность - свойство информации не иметь скрытых ошибок, т.е. отражать истинное положение дел.

3. Актуальность - соответствие информации текущему моменту времени. С течением времени информация стареет.

4. Ценность - свойство информации, отражающее в какой степени она способствует достижению целей и задач ее потребителя.

5. Защищенность - свойство, характеризующее невозможность несанкционированного использования или изменения.

6. Эргономичность - свойство, характеризующее удобство формы или объема информации с точки зрения данного потребителя.

3. Классификация информации

Важным понятием при работе с информацией является классификация объектов.

Классификация - система распределения объектов (предметов, явлений, процессов, понятий) по классам в соответствии с определенным признаком.

Под объектом понимается любой предмет, процесс, явление материального или нематериального свойства. Система классификации позволяет сгруппировать объекты и выделить определенные классы, которые будут характеризоваться рядом общих свойств.

Классификация объектов - это процедура группировки на качественном уровне, направленная на выделение однородных свойств.

Применительно к информации как к объекту классификации, выделенные классы называют информационными объектами (ИО).

Пример. Всю информацию об университете можно классифицировать по многим многочисленным информационным объектам, которые будут характеризоваться общими свойствами:

информация о студентах – в виде ИО «Студент»;

информация о преподавателях – в виде ИО «Преподаватель»;

информация о факультетах – в виде ИО «Факультет» и т.п.

Свойства ИО определяются информационными параметрами, называемыми реквизитами (представляются числовыми данными: вес, стоимость, год или признаками: цвет, фамилия).

Реквизит – логически неделимый информационный элемент, описывающий определенное свойство объекта, процесса, явления.

Пример. Информация о каждом студенте в отделе кадров университета систематизирована и представлена посредством одинаковых реквизитов: фамилия, имя, отчество, пол, год рождения, место рождения, адрес проживания, факультет и т.д.

4. Кодирование информации. Единицы измерения информации

Информация может быть представлена в различных формах - звуковая, печатная и т. д.

В процессе обмена информацией часто приходится производить операции кодирования и декодирования информации.

Кодирование - это операция преобразования знаков или группы знаков одной знаковой системы в знаки или группы знаков другой знаковой системы:

- при вводе знака в компьютер путем нажатия соответствующей клавиши на клавиатуре происходит его *кодирование*, т.е. преобразование во внутренний компьютерный код,

- и наоборот, при выводе знака на экран монитора или принтер происходит обратный процесс - *декодирование*: знак из компьютерного кода преобразуется в его графическое изображение.

Средством кодирования служит таблица соответствия знаковых систем, которая устанавливает взаимно однозначное соответствие между знаками или группами знаков двух различных знаковых систем.

Компьютер может обрабатывать информацию следующих видов:

- числовую,
- текстовую,
- графическую,
- видео,
- звуковую.

Все эти виды информации преобразуются в последовательности электрических импульсов: есть импульс (1), нет импульса (0), т.е. в последовательности нулей и единиц. Такое кодирование информации легко реализуется технически и называется двоичным кодированием.

Существует много кодов: двоичный код обмена информации ДКОИ, восьмеричный код обмена информацией КОИ-8, в современных персональных компьютерах информация представляется в коде ASCII – американский стандартный код для обмена информацией. В России используется так называемая альтернативная ASCII кодировка, содержащая символы русского алфавита.

При вводе информации в компьютер каждый символ-буква русского или латинского алфавита, цифра, знак пунктуации или действия кодируется определенной последовательностью из восьми двоичных цифр в соответствии с таблицей кодирования.

Единицы измерения объема информации

Разные люди, получив одно и тоже сообщение, по-разному оценивают количество информации, содержащееся в нем. Оно зависит от того, насколько ново это сообщение для получателя. При этом подходе трудно выделить критерии, по которым можно было вывести единицу измерения информации.

2. Измерение информации в технике (информация – любая хранящаяся, обрабатываемая или передаваемая последовательность знаков).

А) В технике часто используют способ определения количества информации называемый объемным. Он основан на подсчете числа символов в сообщении, т.е. связан с его длиной и не зависит от содержания.

Б) В вычислительной технике (ВТ) применяют две стандартные единицы измерения:

бит (binary digit) и байт (bate). 1 байт = 8 бит

Бит – минимальная единица измерения информации, которая представляет собой двоичный знак двоичного алфавита {0;1}.

Байт – единица количества информации в СИ, представляющая собой восьмиразрядный двоичный код, с помощью которого можно представить один символ.

Информационный объем сообщения (информационная емкость сообщения) – количество информации в сообщении, измеренное в стандартных единицах или производных от них (Кбайтах, Мбайтах и т.д.).

1 байт = 8 бит

1 Кбайт = 2¹⁰ байт = 1024 байт

1 Мбайт = 2¹⁰ Кбайт = 2²⁰ байт

1 Гбайт = 2¹⁰ Мбайт = 2³⁰ байт

В теории информации количеством информации называют числовую характеристику сигнала, не зависящую от его формы и содержания, и характеризующую неопределенность, которая исчезнет после получения сообщения в виде данного сигнала. В этом случае количество информации зависит от вероятности получения сообщения о том или ином событии.

Для абсолютно достоверного события (событие обязательно произойдет, поэтому его вероятность равна 1) количество информации в сообщении о нем равно 0. Чем неожиданнее событие, тем больше информации он несет.

Лишь при равновероятных событиях: ответ «да» или «нет», несет 1 бит.

3.1. Лекция № 3 (2 часа)

Тема: Технические средства реализации информационных процессов

3.1.1. Вопросы лекции:

1. Классификация ЭВМ

2. Основные типы компьютеров. Конфигурации персональных компьютеров

3. Основные принципы функционирования ПК

4. Состав типового компьютера

3 Краткое содержание вопросов

1. Классификация ЭВМ.

Общепринятой классификацией компьютеров является классификация по поколениям ЭВМ, в основе которой лежит элементная база.

Первое поколение - электронные вакуумные лампы (1946-до середины 50-х годов XX века);

Второе поколение - полупроводниковые приборы, транзисторы (до середины 60-х годов XX века);

Третье поколение - интегральные схемы на полупроводниковых элементах (до конца 70-х годов);

Четвертое поколение - сверхбольшие интегральные схемы (с начала 80-х годов по настоящее время).

Пятое поколение отличительными чертами ЭВМ этого поколения являются новые технологии производства, переход к новым многопроцессорным архитектурам, новые способы ввода-вывода, искусственный интеллект и т.д.

Классификация компьютеров по назначению

Специализированные - предназначены для решения узкого круга специальных задач, например по управлению конкретными техническими устройствами, технологическими процессами (станками с числовым программным управлением, роботами и т.д.).

Универсальные - используются в различных сферах человеческой деятельности для решения самых разнообразных задач: инженерно-технических, экономических, математических, информационно-поисковых и других.

2. Основные типы компьютеров. Конфигурации персональных компьютеров

Персональные компьютеры представляют наиболее многочисленный и разнообразный по составу класс ЭВМ. Они используются при решении самых разных задач не только профессиональными программистами, но и специалистами других областей знаний и деятельности.

К ПК (или ПЭВМ) относится ЭВМ, управляемая одним пользователем и предоставляющая пользователю в каждом сеансе работы все свои ресурсы. К особенностям, отличающим ПК от других ЭВМ, следует отнести:

1. Универсальный характер использования, в соответствии с которым на ПК могут решаться экономические, научные, производственно-технические, конструкторско-технологические и другие задачи в различных сферах человеческой деятельности.

2. Модульный характер построения архитектуры ПК, позволяющий формировать техническую конфигурацию, определять состав внутренних и внешних устройств ПК в зависимости от характера решаемых задач, требований пользователя и финансовых возможностей.

3. Развитость и разнообразие программного обеспечения (ПО), направленные на решение задач из различных областей знаний и деятельности человека.

4. Небольшие габариты, высокая надежность работы, отсутствие специальных требований к условиям эксплуатации и наличие <дружественного> человеко-машинного интерфейса, дающие возможность устанавливать ПК на рабочие места пользователя.

Первоначально основным признаком ПК служило наличие в нем микропроцессора (МП), выполненного в виде одной микросхемы. В настоящее время этот признак перестал быть определяющим, так как МП используются во всех классах ЭВМ.

ПК классифицируются по следующим признакам:

1. По размеру ПК делятся на стационарные (Desktop) и переносные ПК. В состав переносных ПК включаются портативные (Laptop), блокнотного типа (Notebook) и карманные (Palmtop) ПК.

2. По типу используемых МП различают ПК, построенные на процессорах с расширенной системой команд - CISC-процессорах (CISC - Complete Instruction Set Computer), и ПК, основанные на процессорах с сокращенным набором команд - RISC-процессорах (RISC - Reduce Instruction Set Computer).

К первой группе ПК относятся ПК с Intel-совместимой архитектурой. Во вторую группу входят ПК Macintosh на МП Power PC (более ранние модели ПК Macintosh

выпускались на базе МП Motorola 680x0), компьютеры SUN с процессорами SPARC-архитектуры, компьютеры DEC с МП Alpha и другие менее распространенные ПК.

Необходимо обратить внимание, что приведенные ПК представляют не отдельные модели, а программно совместимые <сверху-вниз> семейства поколений компьютеров.

3. Классификация ПК по поколениям МП включает для Intel-совместимых ПК следующие поколения: 8086/8088, 80286, 80386, 80486, Pentium, Pentium II, Pentium III, Pentium IV. В каждом поколении имеются различные модификации моделей МП, различающиеся структурой, техническими характеристиками и ценой.

4. Классификация ПК по назначению устанавливает четыре группы ПК:

- массовые ПК, к категории которого также относятся рабочие станции клиентов вычислительных сетей;
- специализированные ПК, ориентированные на работу в конкретных предметных областях, например в области медицины, дизайнерской работе и т.д.
- мультимедийные ПК;
- компьютеры, выполняющие функции серверов в вычислительных сетях.

3. Основные принципы функционирования ПК

ЭВМ определяется как комплекс взаимосвязанных программно-управляемых технических устройств, предназначенных для автоматизированной обработки данных с целью получения результатов решения вычислительных и информационных задач.

Основные принципы функционирования компьютера были сформулированные в 1945 году Джоном фон Нейманом.

1. В основе работы ЭВМ лежит программный принцип, согласно которому все вычисления выполняются путем последовательного выполнения команд программы ЭВМ.

2. Принцип хранимой программы означает, что программы и данные во время выполнения программы хранятся в одном адресном пространстве в оперативной памяти и различаются не по способу кодирования, а по способу использования. Программа может выступать также в качестве исходных данных (самомодифицируемые программы).

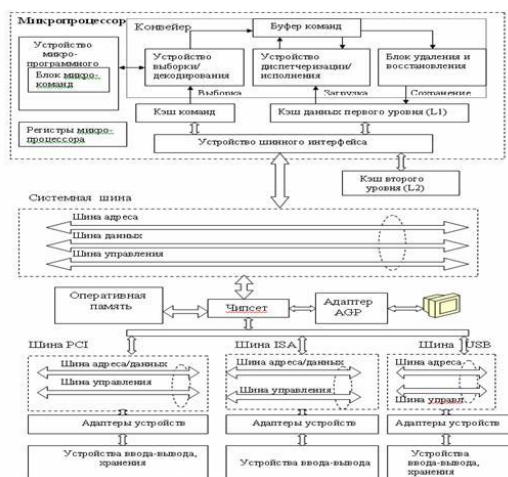
3. Использование двоичного кодирования при хранении и обработке данных в ЭВМ. Отдельные разряды двоичного числа объединяются в более крупные единицы, называемые словами.

4. Информация размещается в ячейках различных запоминающих устройств. Каждая ячейка памяти имеет адрес, по которому происходят запись или считывание слов данных и программ.

4. Состав типового компьютера

Структуру ПК рассмотрим на примере обобщенной схемы технической конфигурации ПК, представленной на рис. 7. Устройства, составляющие внутреннюю конфигурацию ПК, располагаются на системной плате.

К основным устройствам внутренней конфигурации относят: микропроцессор (МП), оперативную память, основной набор микросхем (ChipSets, чипсет), КЭШ-память, а также интерфейсные шины, используемые для связи устройств между собой: системную шину, шины ISA (Industry Standard Architecture - архитектура промышленного стандарта), PCI (Peripheral Component Interconnect - соединение внешних устройств), USB (Universal Serial Bus - универсальная последовательная шина), AGP (Advanced Graphic Port - расширенный графический порт), генератор тактовых импульсов (ГТИ), адаптеры и порты ввода-вывода. Внешние или периферийные устройства ПК подключаются к интерфейсным шинам через адаптеры (или контроллеры) и предназначены для организации интерфейса ПК с пользователем, осуществления операций ввода-вывода и хранения данных. Во многом, благодаря разнообразным внешним устройствам, обеспечивается универсальность использования ПК в различных областях деятельности человека. В зависимости от типа устройства,



Внешние устройства ПК.

Внешние устройства делятся на три группы:

- устройства ввода данных;
- устройства вывода данных;
- устройства хранения данных.

К устройствам ввода данных относят: клавиатуру, сканер, графический планшет, специальные манипуляторы: мышь, джойстик, трекбол, татчпад.

Устройства вывода включают: монитор, принтер, графопостроитель

Внешние устройства хранения данных составляют накопители на: жестком диске, компакт-диске, DVD, гибких дисках, магнитооптических устройствах. К этой же группе устройств относятся внешняя флэш-память и стример.

Состав и функции микропроцессора ПК.

МИКРОПРОЦЕССОР - самостоятельное или входящее в состав микроЭВМ устройство обработки информации, выполненное в виде одной или нескольких больших интегральных схем. Микропроцессор и средства вычислительной техники и автоматики на их основе применяются в системах автоматического управления, технологического и контрольно-испытательного оборудования, транспортных средств, бытовых приборов и др.

Микропроцессор - это интегральная схема, смонтированная кремниевой пластине. Процессор содержит тысячи, или даже миллионы транзисторов, связанных между собой сверхтонкими алюминиевыми соединительными каналами, обеспечивающими их взаимодействие при записи и обработке данных, позволяя микропроцессору выполнять множество полезных функций. Конкретные задачи микропроцессора определяются программным обеспечением.

Микропроцессоры используются, прежде всего, в качестве "мозга" персональных компьютеров, но благодаря им становятся "разумными" и многие другие устройства. Например, оснащенный микропроцессором телефонный аппарат имеет возможность ускоренного и повторного набора номера, домашний термостат автоматически отключается по ночам, автомобиль становится безопаснее и потребляет меньше горючего.

Технологии, связанные с компьютерными микросхемами, используется сейчас повсеместно - от космических кораблей и компьютеров до светофоров и кофемолок. Если устройству можно "приказать" сделать что-нибудь, или запрограммировать его свойства, то в него наверняка встроена компьютерная микросхема.

Назначение и сложность микросхем может сильно отличаться. Один из самых совершенных типов микросхем - это микропроцессоры. Встроенные в них схемы могут выполнять сотни миллионов команд в секунду.

Микропроцессор характеризуется:

- тактовой частотой, определяющей максимальное время выполнения переключения элементов в ЭВМ;
- разрядностью, т.е. максимальным числом одновременно обрабатываемых двоичных разрядов

Разрядность МП обозначается $m/n/k$ и включает:

m - Разрядность внутренних регистров, определяет принадлежность к тому или иному классу процессоров;

n - Разрядность шины данных, определяет скорость передачи информации;

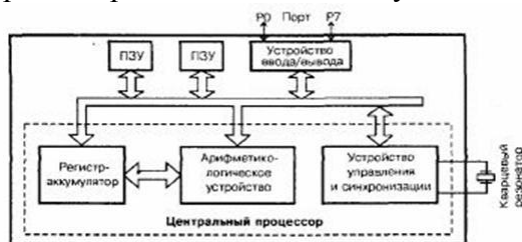
k - Разрядность шины адреса, определяет размер адресного пространства.

Например, МП i8088 характеризуется значениями $m/n/k=16/8/20$;

- архитектурой. Понятие архитектуры микропроцессора включает в себя систему команд и способы адресации, возможность совмещения выполнения команд во времени, наличие дополнительных устройств в составе микропроцессора, принципы и режимы его работы выделяют понятия микроархитектуры и макроархитектуры.

Архитектура типового микропроцессора.

Микропроцессор или микрокомпьютер является практически законченной системой управления. Он имеет сложную архитектуру и представляет собой сверхбольшую интегральную схему, выполненную, как правило, на одном полупроводниковом кристалле. Различные типы микропроцессоров отличаются типом и размером памяти, набором команд, скоростью обработки данных, количеством входных и выходных линий, разрядностью данных. В самом общем виде структурная схема микропроцессора может иметь следующий вид рисунке ниже.



Центральный процессор (CPU) является обязательным узлом любого микропроцессорного устройства, его ядром. В его состав входит: арифметико-логическое устройство (АЛУ); регистр-аккумулятор; логические устройства управления и синхронизации; внутренняя шина.

Арифметико-логическое устройство выполняет арифметические или логические операции над данными, представленными в двоичном или двоично-десятичном коде. Результат выполнения операции сохраняется в так называемом регистре-аккумуляторе. Регистр-аккумулятор представляет собой ячейки оперативной памяти, но, в отличие от ОЗУ, обмен информацией производится более короткими командами, т.е. регистр-аккумулятор является наиболее быстродействующим устройством памяти микропроцессора.

Устройство управления и синхронизации применяется для управления другими узлами микропроцессора, обеспечивая выполнение необходимых задач в соответствии с программой, хранимой в ПЗУ. Узел синхронизации обеспечивает синхронную работу всех узлов с помощью импульсов синхронизации и других управляющих сигналов. В состав устройства управления и синхронизации входит тактовый генератор и формирователь тактовых импульсов. Для генерации импульсов синхронизации используется кварцевый генератор, имеющий внешний кварцевый резонатор. Частота тактового генератора определяет быстродействие микропроцессора.

Связь между различными элементами микропроцессора осуществляется с помощью внутренней шины. Шина - это группа проводников, используемых в качестве линии связи для передачи цифровой информации. В микропроцессоре имеется три основных вида шин: это шина данных, адресная шина и шина управления.

Шина данных обеспечивает передачу данных между узлами процессора. Адресная шина используется для передачи адреса ячейки памяти с целью получить данные из постоянного запоминающего устройства или оперативного запоминающего устройства. Шина управления используется для передачи управляющих сигналов от микропроцессора к другим элементам системы.

Постоянное запоминающее устройство (ПЗУ) используется для хранения постоянной информации, которая вводится в него на этапе производства микропроцессора и не может быть изменена. Это значит, что записанные на заводе-изготовителе данные сохраняются неизменными при выключении питания микропроцессора. ПЗУ расположено на кристалле микропроцессора и состоит из большого количества ячеек. Каждая ячейка памяти имеет свой порядковый номер, называемый адресом. В этих ячейках хранятся коды команд - это и есть управляющая программа, исполняемая микропроцессором во время его работы. Информация вводится в ПЗУ на этапе изготовления микропроцессора, а процедура введения этой информации называется масочным программированием.

Оперативное запоминающее устройство (ОЗУ) используется для временного хранения промежуточных данных. Микропроцессор в процессе работы может изменять эти данные. При выключении питания информация, хранимая временно в ОЗУ, не сохраняется.

Устройство ввода/вывода (интерфейс ввода/вывода) обеспечивает связь с периферийными устройствами - микросхемами, клавиатурой и др. Подключение к внешним устройствам производится через специальные устройства, называемые портами. Они выполнены в виде набора двунаправленных линий. На структурной схеме показан параллельный 8-разрядный порт (выводы 0...7), который можно конфигурировать различным образом. Последовательный порт можно реализовать, используя две линии параллельного порта - одну для передачи, другую для приема необходимых данных. Количество портов может быть любое и зависит от выполняемых микропроцессором задач.

Классификация и характеристика видов памяти и запоминающих устройств ПК:

Жесткий диск

Жесткий диск (накопители на жестких магнитных дисках, НЖМД) - тип постоянной памяти. В отличие от оперативной памяти, данные, хранящиеся на жестком диске, не теряются при выключении компьютера, что делает жесткий диск идеальным для длительного хранения программ и файлов данных, а также самых важных программ операционной системы. Эта его способность (сохранение информации в целостности и сохранности после выключения) позволяет доставать жесткий диск из одного компьютера и вставлять в другой.

Жесткие диски очень надежны для хранения большого объема информации и данных. Внутри запечатанного жесткого диска находятся один или больше несгибающихся дисков, покрытых металлическими частицами. Каждый диск имеет головку (маленький электромагнит), встроенную в шарнирный рычаг, который движется над диском при его вращении. Головка намагничивает металлические частички, заставляя их выстраиваться для представления нулей и единиц двоичных чисел. Моторы,двигающие диск и рычаг, обычно подвергаются износу. Избежать износа удастся только головке, поскольку она никогда не соприкасается с поверхностью диска.

Еще одна функция жесткого диска - симуляция оперативной памяти. Используя секции жесткого диска в качестве виртуальной памяти, Windows может запускать больше программ. Недостаток виртуальной памяти в ее медленности по сравнению с обычной памятью. Если поставить больше, работа компьютера замедляется.

Винчестер, или жесткий диск, - самая важная составляющая компьютера. На нем хранится операционная система, программы и данные. Без операционной системы Windows нельзя запустить компьютер, а без программ - ничего сделать, когда он уже загрузился. Без банка данных придется информацию каждый раз вводить вручную.

Жесткий диск - механическое устройство компьютера, и от него может быть больше проблем, чем от электронных устройств. На самом деле оно очень надежно. Диски собирают в чистых комнатах, в которых воздух постоянно фильтруется, и удаляются частички пыли. Собирают винчестеры из магниточувствительного материала. Перед тем как вынести диски из комнаты, их упаковывают и запечатывают. Если вы откроете свой жесткий диск из любопытства, то можете с ним попрощаться. Чтобы этого не случилось, никогда не делайте этого - их вскрывать нельзя.

4.1. Лекция № 4 (2 часа)

Тема: Программные средства реализации информационных процессов. Системное программное обеспечение.

4.1.1 Вопросы лекции:

1. Понятие программного обеспечения и классификация. Системное программное обеспечение
2. Понятие операционных систем, назначение, функции, классификация
3. Понятие файловой системы
4. Операционная система WINDOWS. Общие сведения о WINDOWS. Интерфейс пользователя WINDOWS. Основы работы в WINDOWS

4.1.2. Краткое содержание вопросов

1. Понятие программного обеспечения и классификация. Системное программное обеспечение

Совокупность программ, предназначенная для решения задач на ПК, называется программным обеспечением. Состав программного обеспечения ПК называют программной конфигурацией.

Программное обеспечение, можно условно разделить на три категории:

- системное ПО (программы общего пользования), выполняющие различные вспомогательные функции, например создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.д.
- прикладное ПО, обеспечивающее выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, обработка информационных массивов и т.д.
- инструментальное ПО (системы программирования), обеспечивающее разработку новых программ для компьютера на языке программирования.



Системное ПО

Это программы общего пользования не связаны с конкретным применением ПК и выполняют традиционные функции: планирование и управление задачами, управления вводом-выводом и т.д.

Другими словами, системные программы выполняют различные вспомогательные функции, например, создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.п.

К системному ПО относятся:

- операционные системы (эта программа загружается в ОЗУ при включении компьютера)
- программы – оболочки (обеспечивают более удобный и наглядный способ общения с компьютером, чем с помощью командной строки DOS, например, Norton Commander)
- операционные оболочки – интерфейсные системы, которые используются для создания графических интерфейсов, мультипрограммирования и т.
- драйверы (программы, предназначенные для управления портами периферийных устройств, обычно загружаются в оперативную память при запуске компьютера)
- утилиты (вспомогательные или служебные программы, которые представляют пользователю ряд дополнительных услуг)

К утилитам относятся:

- диспетчеры файлов или файловые менеджеры
- средства динамического сжатия данных (позволяют увеличить количество информации на диске за счет ее динамического сжатия)
- средства просмотра и воспроизведения
- средства диагностики; средства контроля позволяют проверить конфигурацию компьютера и проверить работоспособность устройств компьютера, прежде всего жестких дисков
- средства коммуникаций (коммуникационные программы) предназначены для организации обмена информацией между компьютерами
- средства обеспечения компьютерной безопасности (резервное копирование, антивирусное ПО).

Необходимо отметить, что часть утилит входит в состав операционной системы, а другая часть функционирует автономно. Большая часть общего (системного) ПО входит в состав ОС. Часть общего ПО входит в состав самого компьютера (часть программ ОС и контролирующих тестов записана в ПЗУ или ППЗУ, установленных на системной плате). Часть общего ПО относится к автономным программам и поставляется отдельно.

Прикладное ПО

Прикладные программы могут использоваться автономно или в составе программных комплексов или пакетов. Прикладное ПО – программы, непосредственно обеспечивающие выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, создание электронных таблиц и т.д.

Пакеты прикладных программ – это система программ, которые по сфере применения делятся на проблемно – ориентированные, пакеты общего назначения и интегрированные пакеты. Современные интегрированные пакеты содержат до пяти функциональных компонентов: тестовый и табличный процессор, СУБД, графический редактор, телекоммуникационные средства.

К прикладному ПО, например, относятся:

- Комплект офисных приложений MS OFFICE
- Бухгалтерские системы
- Финансовые аналитические системы
- Интегрированные пакеты делопроизводства
- CAD – системы (системы автоматизированного проектирования)
- Редакторы HTML или Web – редакторы

- Браузеры – средства просмотра Web - страниц
- Графические редакторы
- Экспертные системы
- И так далее.

2. Понятие операционных систем, назначение, функции, классификация

Операционная система составляет основу программного обеспечения ПК. Операционная система представляет комплекс системных и служебных программных средств, который обеспечивает взаимодействие пользователя с компьютером и выполнение всех других программ.

С одной стороны, она опирается на базовое программное обеспечение ПК, входящее в его систему BIOS, с другой стороны, она сама является опорой для программного обеспечения более высоких уровней – прикладных и большинства служебных приложений.

Для того чтобы компьютер мог работать, на его жестком диске должна быть установлена (записана) операционная система. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ. Этот процесс называется загрузкой операционной системы.

Операционные системы различаются особенностями реализации алгоритмов управления ресурсами компьютера, областями использования.

Так, в зависимости от алгоритма управления процессором, операционные системы делятся на:

- Однозадачные и многозадачные
- Однопользовательские и многопользовательские
- Однопроцессорные и многопроцессорные системы
- Локальные и сетевые.

По числу одновременно выполняемых задач операционные системы делятся на два класса:

- Однозадачные (MS DOS)
- Многозадачные (OS/2, Unix, Windows)

В однозадачных системах используются средства управления периферийными устройствами, средства управления файлами, средства общения с пользователями. Многозадачные ОС используют все средства, которые характерны для однозадачных, и, кроме того, управляют разделением совместно используемых ресурсов: процессор, ОЗУ, файлы и внешние устройства.

В зависимости от областей использования многозадачные ОС подразделяются на три типа:

- Системы пакетной обработки (ОС ЕС)
- Системы с разделением времени (Unix, Linux, Windows)
- Системы реального времени (RT11)

Системы пакетной обработки предназначены для решения задач, которые не требуют быстрого получения результатов. Главной целью ОС пакетной обработки является максимальная пропускная способность или решение максимального числа задач в единицу времени.

Эти системы обеспечивают высокую производительность при обработке больших объемов информации, но снижают эффективность работы пользователя в интерактивном режиме.

В системах с разделением времени для выполнения каждой задачи выделяется небольшой промежуток времени, и ни одна задача не занимает процессор надолго. Если этот промежуток времени выбран минимальным, то создается видимость одновременного выполнения нескольких задач. Эти системы обладают меньшей пропускной

способностью, но обеспечивают высокую эффективность работы пользователя в интерактивном режиме.

Системы реального времени применяются для управления технологическим процессом или техническим объектом, например, летательным объектом, станком и т.д.

По числу одновременно работающих пользователей на ЭВМ ОС разделяются на однопользовательские (MS DOS) и многопользовательские (Unix, Linux, Windows 95 - XP)

В многопользовательских ОС каждый пользователь настраивает для себя интерфейс пользователя, т.е. может создать собственные наборы ярлыков, группы программ, задать индивидуальную цветовую схему, переместить в удобное место панель задач и добавить в меню Пуск новые пункты.

В многопользовательских ОС существуют средства защиты информации каждого пользователя от несанкционированного доступа других пользователей.

Многoproцессорные и однопроцессорные операционные системы. Одним из важных свойств ОС является наличие в ней средств поддержки многопроцессорной обработки данных. Такие средства существуют в OS/2, Net Ware, Windows NT. По способу организации вычислительного процесса эти ОС могут быть разделены на асимметричные и симметричные.

Одним из важнейших признаков классификации ЭВМ является разделение их на локальные и сетевые. Локальные ОС применяются на автономных ПК или ПК, которые используются в компьютерных сетях в качестве клиента.

В состав локальных ОС входит клиентская часть ПО для доступа к удаленным ресурсам и услугам. Сетевые ОС предназначены для управления ресурсами ПК включенных в сеть с целью совместного использования ресурсов. Они представляют мощные средства разграничения доступа к информации, ее целостности и другие возможности использования сетевых ресурсов.

3. Понятие файловой системы

Все современные ОС обеспечивают создание файловой системы, которая предназначена для хранения данных на дисках и обеспечения доступа к ним.

Основные функции файловой системы можно разделить на две группы:

Функции для работы с файлами (создание, удаление, переименование файлов и т.д.)

Функции для работы с данными, которые хранятся в файлах (запись, чтение, поиск данных и т.д.)

Известно, что файлы используются для организации и хранения данных на машинных носителях. Файл – это последовательность произвольного числа байтов, обладающая уникальным собственным именем или поименованная область на машинных носителях.

Структурирование множества файлов на машинных носителях осуществляется с помощью каталогов, в которых хранятся атрибуты (параметры и реквизиты) файлов. Каталог может включать множество подкаталогов, в результате чего на дисках образуются разветвленные файловые структуры. Организация файлов в виде древовидной структуры называется файловой системой.

Принцип организации файловой системы – табличный. Данные о том, в каком месте на диске записан файл, хранятся в таблице размещения файлов (File Allocation Table, FAT).

Эта таблица размещается в начале тома. В целях защиты тома на нем хранятся две копии FAT. В случае повреждения первой копии FAT дисковые утилиты могут воспользоваться второй копией для восстановления тома.

По принципу построения FAT похожа на оглавление книги, так как операционная система использует ее для поиска файла и определения кластеров, которые этот файл занимает на жестком диске.

Наименьшей физической единицей хранения данных является сектор. Размер сектора 512 байт. Поскольку размер FAT – таблицы ограничен, то для дисков, размер которых превышает 32 Мбайт, обеспечить адресацию к каждому отдельному сектору не представляется возможным.

4. Операционная система WINDOWS. Общие сведения о WINDOWS. Интерфейс пользователя WINDOWS. Основы работы в WINDOWS

Операционная система Windows - это современная многозадачная многопользовательская 32 - разрядная ОС с графическим интерфейсом пользователя. Операционные системы семейства Windows являются наиболее распространенными ОС, которые установлены в домашних и офисных ПК.

Графическая оболочка ОС Windows обеспечивает взаимодействие пользователя с компьютером в форме диалога с использованием ввода и вывода на экран дисплея графической информации, управления программами с помощью пиктограмм, меню, окон, панелей (управления, задач, инструментов) и других элементов управления.

Основными элементами графического интерфейса Windows являются: Рабочий стол, Панель задач с кнопкой Пуск. Так как в Windows применен графический пользовательский интерфейс, то основным устройством управления программами является манипулятор мышь.

Основными элементами графического интерфейса Windows являются:

- Рабочий стол с пиктограммами
- Панель задач, на которой размещаются программные кнопки, индикаторы, Панель быстрого запуска

- Главное меню (кнопка Пуск)
- Контекстное меню (отображается при щелчке правой кнопкой мыши по выбранному объекту)

В ОС Windows применяются четыре типа меню (меню – это список команд, выводимых на экран и предлагаемых пользователю для выбора):

- Главное меню (открывается кнопкой Пуск)
- Строка меню в окнах приложения (все программы, входящие в стандартный пакет поставки Windows, имеют строку меню)
- Системное меню в окнах приложения (для изменения размеров окна и его положения)
- Контекстное меню

Работа с файлами

Все файлы, документы и программы в Windows хранятся в папках. В электронной папке, как правило, хранят файлы, сгруппированные по какому-либо признаку, типу и другие папки.

Папка – это контейнер для программ и файлов в графических интерфейсах пользователя, отображаемый на экране с помощью значка, имеющего вид канцелярской папки. Windows предоставляет средства для управления файлами и папками.

К таким средствам относятся программа Проводник и окно Мой компьютер. Приложение Проводник является главным инструментом Windows для просмотра файлов и папок, хранящихся на жестких и гибких дисках и других носителях информации.

Проводник отображает иерархическую структуру файлов, папок и дисков на ПК. В левой части проводника Windows использует иерархическое представление папок, файлов и других ресурсов, подключенных к компьютеру или сети.

Мой компьютер – программа, используемая для работы с файлами и папками, хранящимися на дисках компьютера. Мое сетевое окружение – программа, используемая для работы с сетевыми ресурсами в рабочей группе.

5.1. Лекция № 5 (2 часа)

Тема: Основные понятия и принципы моделирования

5.1.1 Вопросы лекции:

1. Понятие модели и моделирования
2. Назначение моделей
3. Основные этапы построения моделей
4. Понятие формализации

5.1.2. Краткое содержание вопросов

1. Понятие модели и моделирования

Объекты и явления окружающего мира, даже самые на взгляд простые, на самом деле сложны. Чтобы понять, как объект или явление действует, приходится рассматривать их модели.

Модель – упрощенное представление, аналог реального объекта, процесса или явления. При построении модели сам объект называют **оригиналом**, или **прототипом**. Например, детская игрушка представляет реальный объект окружающего мира, но зачастую только одно или несколько свойств – внешний вид. Для любого объекта может существовать множество моделей, различных по сложности и степени сходства с оригиналом.

Модель необходима для того, чтобы:

- понять, как устроен реальный объект: каковы его структура, основные свойства, законы развития и взаимодействия с окружающим миром;
- научиться управлять объектом и процессом: определить наилучшие способы управления при заданных целях и критериях (оптимизация);
- прогнозировать прямые или косвенные последствия реализации заданных способов и форм воздействия на объект.

Моделирование – построение и изучение моделей в целях получения новых знаний или дальнейшего совершенствования характеристик объектов исследования.

Моделирование – это метод научного познания объективного мира с помощью моделей. Модель строится таким образом, чтобы она наиболее полно воспроизводила необходимые для изучения качества объекта. Модель должна быть проще объекта и удобнее для изучения. Таким образом, для одного и того же объекта могут существовать различные модели, классы моделей, соответствующие различным целям его изучения.

2. Назначение моделей

Моделировать можно внешний вид, структуру, поведение объекта, а также все возможные их комбинации. Это называется **аспект** моделирования.

Структурой объекта называют совокупность его элементов, а также существующих между ними связей.

Поведением объекта называют изменение его внешнего вида и структуры с течением времени в результате взаимодействия с другими объектами.

В процессе моделирования каждый аспект раскрывается через совокупность свойств, но отражаются только существенные с точки зрения целей моделирования.

Каждый аспект моделирования можно охарактеризовать набором свойств:

- внешний вид – набором признаков;
- структуру – перечнем элементов и указанием отношений между ними;
- поведение – изменением внешнего вида и структуры с течением времени.

Некоторые свойства объекта могут быть выражены числовыми значениями. Такие величины носят название **параметров модели**.

3. Основные этапы построения моделей

Основные этапы построения моделей

Процесс моделирования можно разбить на следующие этапы:

- определение цели моделирования;
- анализ объекта и выделение всех известных его свойств;
- анализ выделенных свойств с точки зрения цели моделирования и определение, какие из них следует считать существенными;
- выбор формы представления модели;
- формализация;
- анализ полученной модели на непротиворечивость;
- анализ адекватности полученной модели объекту и цели моделирования.

Если условия моделирования позволяют, то рекомендуется построить несколько моделей с разными наборами существенных свойств и затем оценить их на адекватность объекту и цели моделирования.

Классификация моделей:

Наиболее распространенными классификационными признаками для моделей являются:

- область использования;
- временной фактор (динамика);
- область знаний;
- способ представления.

По области **использования** можно выделить следующие виды.

Учебные модели. К ним можно отнести обучающие программы, тренажеры.

Опытные модели – измененные в размерах копии проектируемых объектов, на основе которых исследуются и прогнозируются их будущие характеристики. К ним можно отнести модели корабля в бассейне, автомобиля или самолета в аэродинамической трубе и др.

Научно-технические модели используются для исследования процессов и явлений (стенд для тестирования телетехники, громоотвод и др.).

Игровые модели – это различные игры (военные, спортивные, деловые, экономические), проигрывающие реальные и потенциальные ситуации.

Имитационные модели представляют процессы реальной жизни с различной степенью точности. Эксперименты при различных исходных данных позволяют сделать выводы. Этот метод подбора правильного решения получил название «метод проб и ошибок».

По учету **фактора времени** выделяют следующие модели.

Статические, представляющие собой единовременный срез информации по данному объекту.

Динамические, представляющие картину изменения объекта во времени.

Один и тот же объект можно охарактеризовать и статической, и динамической моделями.

По области **знаний** или деятельности модели делятся на *биологические, социологические, экономические, исторические* и т.п.

По **способу представления** («из чего сделаны») модели делят на две группы: *материальные* и *абстрактные* (нематериальные). Последняя категория, в свою очередь, подразделяется на *мысленные* (вербальные) и *информационные* модели.

Обе группы содержат информацию об исходном объекте: цвет, форма, пропорция и т.п. В материальной модели эта информация имеет реальное воплощение, ее можно получить с помощью органов чувств и измерительными приборами. Исследователь материально воздействует на модель. Материальные модели могут быть *физическими, аналоговыми* и *пространственными*.

Физическое моделирование предназначено для воспроизводства динамики процессов, происходящих в реальных объектах (испытания объектов в аэротрубе).

Аналоговое моделирование основано на аналогии математического описания моделей и объектов, т.е. связано с использованием материальных моделей, имеющих иную физическую природу, чем изучаемый объект, но описывающихся теми же математическими соотношениями, что и изучаемый объект.

В пространственном моделировании используются модели, предназначенные для восприятия пространственных или геометрических свойств изучаемого объекта (макеты, глобусы).

В нематериальной модели та же информация представлена в абстрактной форме (мысль, формула, чертеж, схема) и основана на умозрительной связи между объектом и моделью. Этот вид моделей может быть *формализованным* и *неформализованным*.

В формализованном моделировании моделью служат системы знаков (знаковое моделирование) или образов (образное моделирование), вместе с коэффициентами задаются правила их преобразования и представления.

К *знаковому* моделированию относится математика, позволяющая для различных явлений применить одинаковые математические описания в виде совокупности формул, уравнений, подчиненных правилам логики и математики. В математических методах широко применяются *аналитические* и *статистические* модели. Каждый из этих типов имеет свои преимущества и недостатки.

Аналитические модели учитывают меньшее число факторов, всегда включают допущения и упрощения. Но результаты расчета отчетливее отображают основные закономерности явления.

Статистические модели более точны и подробны, позволяют учесть большее число факторов. К их недостаткам можно отнести громоздкость, плохую обзорность и трудность поиска оптимального решения.

Наилучший результат получается при совместном применении аналитических и статистических моделей.

В *образном* моделировании модели строятся на наглядных элементах (фотография, рисунок, фильм или звукозапись). Анализ образных моделей осуществляется мысленно, поэтому они могут быть отнесены к формальному моделированию (например, столкновение двух молекул, как соударение шаров). В физике широко используются «мысленные эксперименты».

Если отображение реальной действительности точно не зафиксировано (модель не формируется), а вместо нее используется нечто мысленное, служащее основой для рассуждения и принятия решения, то такой анализ проблем можно отнести к *неформализованному* моделированию.

Материальная и абстрактная модели взаимно дополняют друг друга.

4. Понятие формализации

В общем виде **формализация** понимается как сведение некоторого содержания (текста, научной теории, воспринимаемых сигналов и проч.) к выбранной форме описания. Например: книга – формализация средствами языковых конструкций мыслей автора; оглавление книги – это формализация ее содержательных частей; научная теория – совокупность текста, формул, графиков, схем, таблиц и т.д.

План действий в результате формализации переводится в алгоритм.

В основе понятия формализации лежит принцип существования принципиальной возможности разделения самого объекта и его обозначения, т.е. суть объекта не меняется от того, как он называется и описывается.

Языки описания бывают естественные и искусственные.

Язык характеризуется набором используемых знаков (слов, фраз, текстов), правилами образования из этих знаков языковых конструкций – синтаксических, семантических и прагматических.

Правила искусственного языка определяются строго и однозначно, поэтому такой язык называется **формализованным**.

6.1. Лекция № 6, 7, 8 (6 часов)

Тема: Алгоритмизация вычислительных процессов

6.1.1. Вопросы лекции:

1. Понятие алгоритмизации вычислительных процессов
2. Понятие алгоритма и его свойства, способы описания алгоритмов
3. Основные типы алгоритмов
 - 3.1. Алгоритм линейной структуры
 - 3.2. Алгоритм ветвящейся структуры
 - 3.3. Алгоритм циклической структуры

6.1.2. Краткое содержание вопросов

1. Понятие алгоритмизации вычислительных процессов

Разработке алгоритма предшествуют такие этапы, как формализация и моделирование задачи. Формализация предполагает замену словесной формулировки решаемой задачи краткими символьными обозначениями, близкими к обозначениям в языках программирования или к математическим. Моделирование задачи является важнейшим этапом, целью которого является поиск общей концепции решения. Обычно моделирование выполняется путем выдвижения гипотез решения задачи и их проверке любым рациональным способом (прикидочные расчеты, физическое моделирование и т.д.). Результатом каждой проверки является либо принятие гипотезы, либо отказ от нее и разработка новой.

При разработке алгоритма используют следующие основные принципы.

Принцип поэтапной детализации алгоритма (другое название - "проектирование сверху-вниз"). Этот принцип предполагает первоначальную разработку алгоритма в виде укрупненных блоков (разбиение задачи на подзадачи) и их постепенную детализацию.

Принцип "от главного к второстепенному", предполагающий составление алгоритма, начиная с главной конструкции. При этом, часто, приходится "достраивать" алгоритм в обратную сторону, например, от середины к началу.

Принцип структурирования, т.е. использования только типовых алгоритмических структур при построении алгоритма. Нетиповой структурой считается, например, циклическая конструкция, содержащая в теле цикла дополнительные выходы из цикла. В программировании нетиповые структуры появляются в результате злоупотребления командой безусловного перехода (GoTo). При этом программа хуже читается и труднее отлаживается.

Говоря о блок-схемах, как о средстве записи алгоритма, можно дать еще один совет по их разработке. Рекомендуется после внесения исправлений в блок-схему аккуратно перерисовывать ее с учетом этих исправлений. Аккуратность записи есть аккуратность мысли программиста. Аккуратно записанный и детализованный алгоритм упрощает его программирование и отладку.

Этапы решения задачи на компьютере.

Решение задачи разбивается на этапы:

1. Постановка задачи
2. Формализация (математическая постановка)
3. Выбор (или разработка) метода решения

4. Разработка алгоритма
5. Составление программы
6. Отладка программы
7. Вычисление и обработка результатов

1. При постановке задачи выясняется конечная цель и вырабатывается общий подход к решению задачи. Выясняется сколько решений имеет задача и имеет ли их вообще. Изучаются общие свойства рассматриваемого физического явления или объекта, анализируются возможности данной системы программирования.

2. На этом этапе все объекты задачи описываются на языке математики, выбирается форма хранения данных, составляются все необходимые формулы.

3. Выбор существующего или разработка нового метода решения (очень важен и, в то же время личностный этап).

4. На этом этапе метод решения записывается применительно к данной задаче на одном из алгоритмических языков (чаще на графическом).

5. Переводим решение задачи на язык, понятный машине.

2. Понятие алгоритма и его свойства, способы описания алгоритмов

"Алгоритм" является фундаментальным понятием информатики. Представление о нем необходимо для эффективного применения вычислительной техники к решению практических задач. Алгоритм - это предписание исполнителю (человеку или автомату) выполнить точно определенную последовательность действий, направленных на достижение заданной цели. Алгоритм - это сформулированное на некотором языке правило, указывающее на действия, последовательное выполнение которых приводит от исходных данных к искомому результату. Значение слова алгоритм очень схоже со значением слов рецепт, процесс, метод, способ. Однако любой алгоритм, в отличие от рецепта или способа, обязательно обладает следующими свойствами.

Свойства алгоритма (отличающие его от любых других предписаний): понятность (для конкретного исполнителя); дискретность (команды последовательны, с точной фиксацией моментов начала и конца выполнения команды); точность (после выполнения каждой команды точно известно, завершено ли исполнение алгоритма или же какая команда должна выполняться следующей); результативность (после конечного числа шагов задача решается или же становится ясно, что процесс решения не может быть продолжен); массовость (алгоритм единым образом применяется к любой конкретной формулировке задачи, для которой он разработан).

1. Дискретность - разбиение алгоритма на ряд отдельных законченных действий - шагов. Выполнение алгоритма разбивается на последовательность законченных действий - шагов. Каждое действие должно быть закончено исполнителем алгоритма прежде, чем он приступит к исполнению следующего действия.

2. Точность - однозначные указания. На каждом шаге однозначно определено преобразование объектов среды исполнителя, полученной на предыдущих шагах алгоритма. Если алгоритм многократно применяется к одному и тому же набору исходных данных, то на выходе он получает каждый раз один и тот же результат. Запись алгоритма должна быть такой, чтобы на каждом шаге его выполнения было известно, какую команду надо выполнять следующей.

3. Понятность - однозначное понимание и исполнение каждого шага алгоритма его исполнителем. Алгоритм должен быть записан на понятном для исполнителя языке.

4. Результативность - обязательное получение результата за конечное число шагов. Каждый шаг (и алгоритм в целом) после своего завершения дает среду, в которой все объекты однозначно определены. Если это по каким-либо причинам невозможно, то алгоритм должен сообщать, что решение задачи не существует. Работа алгоритма должна быть завершена за конечное число шагов. Информатика оперирует только с конечными

объектами и конечными процессами, поэтому вопрос о рассмотрении бесконечных алгоритмов остается за рамками теории алгоритмов.

5. Массовость - применение алгоритма к решению целого класса однотипных задач.

Порядок выполнения алгоритма:

- Действия в алгоритме выполняются в порядке их записи
- Нельзя менять местами никакие два действия алгоритма
- Нельзя не закончив одного действия переходить к следующему

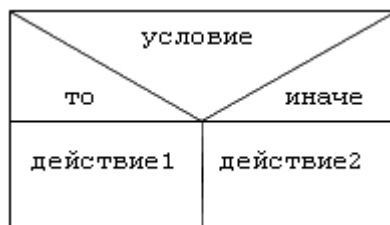
Для записи алгоритмов используются специальные языки:

1. Естественный язык (словесная запись)
2. Формулы
3. Псевдокод
4. Структурограммы
5. Синтаксические диаграммы
6. Графический (язык блок-схем)

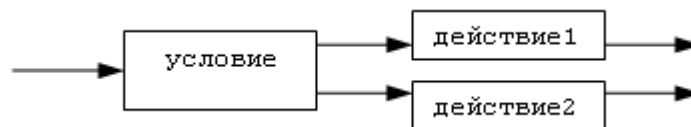
1. Естественный язык:

если условие то действие1 иначе действие2

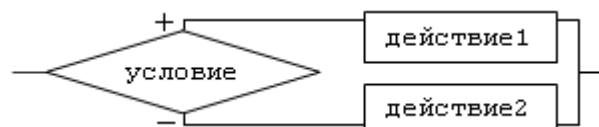
2. Структурограмма:



3. Синтаксическая диаграмма:



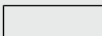
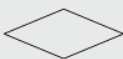
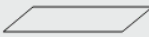
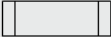
4. Графический язык:



Составление алгоритмов графическим способом подчиняется двум ГОСТам:

1. ГОСТ 19.002-80, соответствует международному стандарту ИСО 2636-73. Регламентирует правила составления блок-схем.

2. ГОСТ 19.003-80, соответствует международному стандарту ИСО 1028-73. Регламентирует использование графических примитивов.

Название	Символ (рисунок)	Выполняемая функция (пояснение)
1. Блок вычислений		Выполняет вычислительное действие или группу действий
2. Логический блок		Выбор направления выполнения алгоритма в зависимости от условия
3. Блоки ввода/вывода		Ввод или вывод данных вне зависимости от физического носителя
		Вывод данных на печатающее устройство
4. Начало/конец (вход/выход)		Начало или конец программы, вход или выход в подпрограмму
5. Предопределенный процесс		Вычисления по стандартной или пользовательской подпрограмме
6. Блок модификации		Выполнение действий, изменяющих пункты алгоритма
7. Соединитель		Указание связи между прерванными линиями в пределах одной страницы
8. Межстраничный соединитель		Указание связи между частями схемы, расположенной на разных страницах

Правила построения блок-схем:

Блок-схема выстраивается в одном направлении либо сверху вниз, либо слева направо

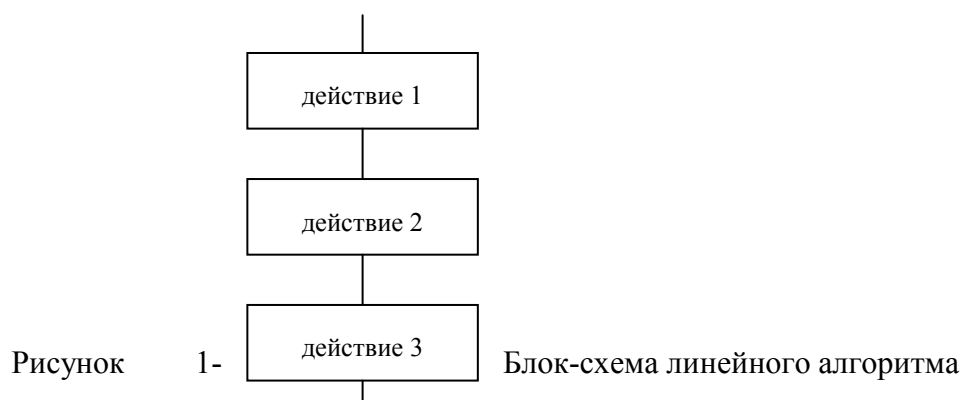
Все повороты соединительных линий выполняются под углом 90 градусов

3. Основные типы алгоритмов

3.1. Алгоритм линейной структуры

Линейным называется **вычислительный процесс**, в котором **предусматривается получение результата путем однократного выполнения последовательности действий при любых значениях исходных данных**. Характерной особенностью линейного вычислительного процесса является то, что направление вычислений не зависит от исходных данных и промежуточных результатов.

Алгоритм линейного вычислительного процесса графически может быть представлен **блоком следования – композицией** (объединением) нескольких следующих друг за другом блоков ПРОЦЕСС (рис. 1).

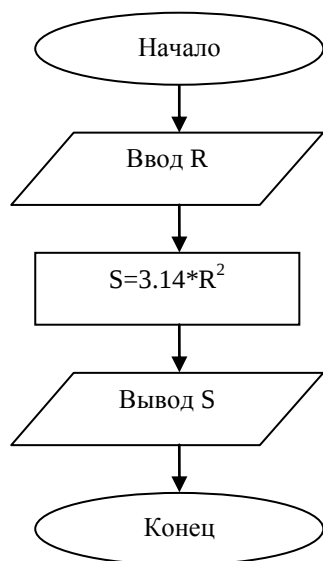


Линии, соединяющие отдельные блоки на блок-схеме алгоритма и указывающие на последовательность действий, называются **линиями потока**. Направление линии потока сверху вниз принимается за основное и стрелкой не обозначается.

Рассмотрим пример задачи, решение которой представляет собой линейный вычислительный процесс, и составим блок-схему алгоритма.

Пример: Найдите площадь круга S при заданном значении радиуса R . Для вычисления площади использовать формулу $S=3,14 \cdot R^2$.

Блок-схема



Как правило, составление алгоритма линейного вычислительного процесса не вызывает затруднений, однако в "чистом виде" такие вычислительные процессы встречаются весьма редко. Чаще всего они являются составной частью более сложных вычислительных процессов. Таким образом:

- линейный вычислительный процесс является наиболее простым при реализации его на ЭВМ.
- блок-схема алгоритма линейного вычислительного процесса представляет собой **композицию** (объединение) нескольких следующих друг за другом блоков ПРОЦЕСС.

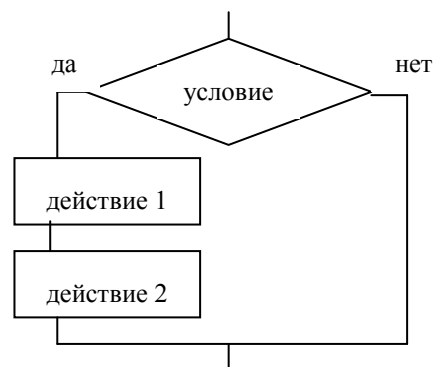
3.2. Алгоритм ветвящейся структуры

Разветвляющимся вычислительным процессом называется процесс, направление вычислений в котором зависит от результата проверки некоторого условия (условий).

Алгоритм разветвляющегося вычислительного процесса предусматривает **выбор** одной из нескольких возможных **альтернатив** (последовательностей действий) в зависимости от значения исходных данных или промежуточных результатов. Каждую из этих последовательностей называют **ветвью алгоритма**. Блок-схема алгоритма разветвляющегося вычислительного процесса содержит, по крайней мере, один блок РЕШЕНИЕ. Направления линий потока сверху вниз и слева направо на блок-схеме принимаются за основные и стрелками не обозначаются.

Графической интерпретацией алгоритма разветвляющегося вычислительного процесса является **блок разветвления алгоритма**, в котором может быть предусмотрен **полный** (рис. 2) или **неполный** (рис. 3) **выбор**.

В блоке разветвления алгоритма с полным выбором в зависимости от результата проверки условия выполняются только действия ветви "да" (т. е., действия 1 и 2) или только действия ветви "нет" (действия 3 и 4). В блоке с неполным выбором в зависимости от результата проверки условия либо выполняются действия какой-либо ветви, либо они игнорируются.



Некоторой разновидностью блока разветвления алгоритма является **блок множественного выбора** (рис. 3). В нем, в зависимости от результатов выбора, выполняется одно из предусмотренных действий.

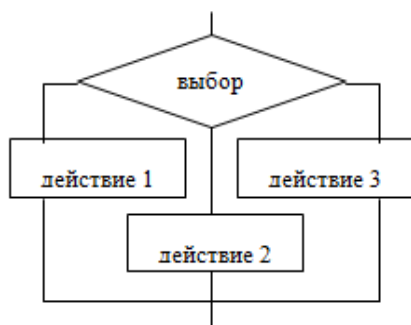
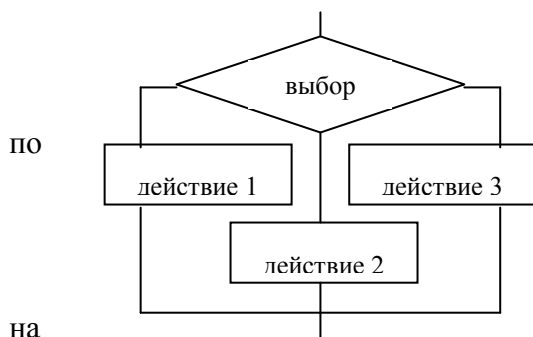


Рисунок 3– Блок множественного выбора

Рассмотрим пример задачи, алгоритм решения которой носит разветвляющийся характер.

Пример 2. Составить схему алгоритма для вычисления значений функции

$$y = \begin{cases} 2x + a, & \text{если } x < 0, \\ x + 3b^2, & \text{если } x \geq 0. \end{cases}$$



Поскольку значение функции вычисляется разным формулам в зависимости от знака аргумента x , в алгоритме вычислительного процесса должна быть предусмотрена проверка знака названного аргумента. Блок-схема алгоритма решения задачи на ЭВМ представлена рис. 4.

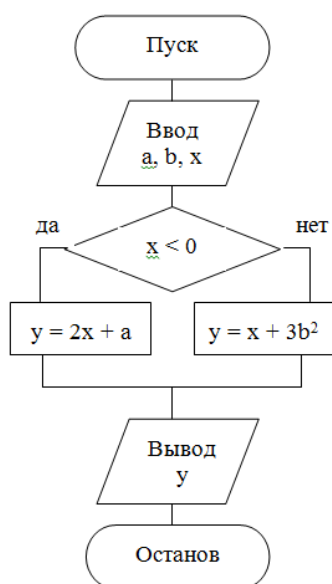


Рисунок 4 – Блок-схема алгоритма

В разветвляющихся вычислительных процессах могут иметь место так называемые "пустые" ветви. В направлении пустых ветвей задача, как правило, не имеет решения (вычислительный процесс не приводит к получению какого-либо промежуточного или конечного результата). Рассмотрим следующий пример.

Пример 3. Составить блок-схему алгоритма для вычисления значений функции

$$y = \begin{cases} x - a, & \text{если } x = 0, \\ x + b^2, & \text{если } 0 < x < 3, \\ x^2 + c, & \text{если } 3 < x < 5. \end{cases}$$

Как видно из условия задачи, переменная x может принимать только значения, находящиеся в диапазоне от 0 до 5. Следовательно, для остальных значений x ветви в алгоритме оказываются пустыми. В блок-схеме алгоритма должны быть предусмотрены действия ЭВМ на случай появления исходных данных (значений x), не описанных логическими условиями. На рис. 6 показан один из возможных способов решения проблемы.

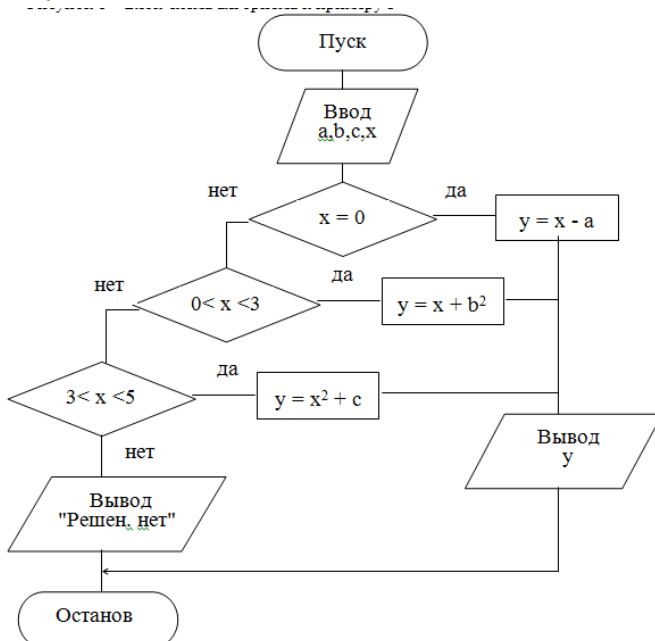


Рисунок 5 – Блок-схема алгоритма к примеру 3

В заключение кратко подведем итог:

1. Алгоритм разветвляющегося вычислительного процесса предусматривает выбор одной из нескольких возможных альтернатив в зависимости от значения исходных данных или промежуточных результатов.
2. В блоке разветвления алгоритма может быть предусмотрен полный или неполный выбор.
3. В разветвляющихся вычислительных процессах могут иметь место "пустые" ветви.

3.3. Алгоритм циклической структуры

Циклическим называется вычислительный процесс, в котором получение результата обеспечивается путем **многократного повторения некоторой последовательности действий**.

Графической интерпретацией алгоритма циклического вычислительного процесса является **блок цикла**. Различают несколько разновидностей блока цикла: **блок цикла с параметром**, **блок цикла с предварительным условием** и **блок цикла с последующим условием**.

Блок-схема **блока цикла с параметром** представлена на рис. 6.

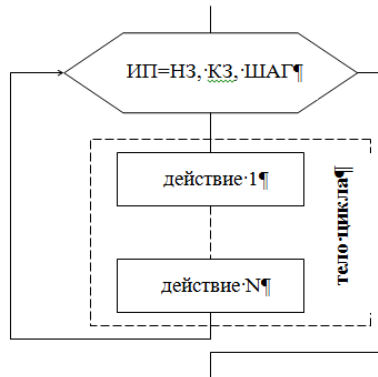


Рисунок 6 – Блок цикла с параметром

На рис. 6 приняты следующие сокращения:

ИП – имя ячейки памяти, в которую заносится значение параметра;

НЗ – начальное значение параметра;

КЗ – конечное значение параметра;

ШАГ – величина приращения параметра после каждого выполнения тела цикла.

Тело цикла представляет собой линейный вычислительный процесс и выполняется столько раз, сколько разных значений примет параметр в заданных пределах от НЗ до КЗ. Блок цикла с параметром относится к циклу **с явно выраженным числом повторений** (число повторений известно заранее). Для таких циклов характерным является то, что задаются:

- **начальное и конечное значения параметра** цикла;
- **закон изменения параметра цикла** при каждом повторном выполнении тела цикла;
- **количество повторных выполнений** тела цикла (вытекает из первых двух пунктов).

Блок цикла с предварительным условием и блок цикла с последующим условием относятся к так называемым **итерационным циклам**. В таких циклических вычислительных процессах число повторений тела цикла заранее не известно. Выход из цикла осуществляется не после того, как цикл повторится заданное число раз, а при выполнении определенного условия, связанного с проверкой значения монотонно

изменяющейся в теле цикла величины. Блок-схема блока цикла с предварительным условием представлена на рис. 8, а блока цикла с последующим условием – на рис. 7.

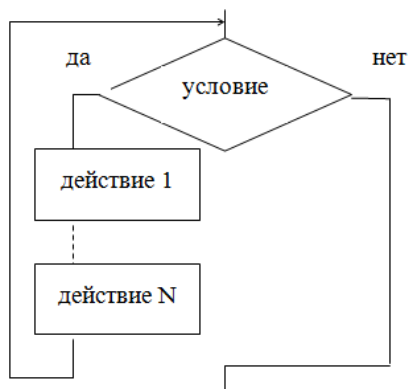


Рисунок 7 – Блок цикла с предварительным условием

Кратко суть алгоритма цикла с предварительным условием можно изложить следующим образом: **пока выполняется условие – повторять действия**. В таких циклах возможны ситуации, когда тело цикла не выполняется ни разу (например, если при первой же проверке не выполняется условие, то сразу происходит выход из цикла).

В цикле с последующим условием (рис. 8) тело цикла выполняется не менее одного раза. При этом **действия**, предусмотренные в теле цикла, **выполняются до тех пор, пока не выполнится заданное условие**.

Рассмотренные блоки циклов позволяют описать **простые** циклические вычислительные процессы. При решении сложных задач может возникнуть необходимость внутри одного цикла организовать дополнительно один или несколько циклов. Такие циклы называются **вложенными**. При этом цикл, внутри которого создается другой цикл, называется **внешним**, а цикл, создаваемый внутри другого – **внутренним**. Правила организации как внешнего, так и внутреннего циклов те же, что и для простых циклов. Параметры внешнего и внутреннего циклов должны быть разными.

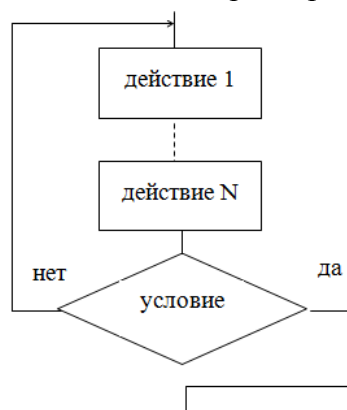


Рисунок 8 - Блок цикла с последующим условием

7.1. Лекция № 9 (2 часа)

Тема: Программные средства реализации информационных процессов.
Инструментальное программное обеспечение

7.1.1. Вопросы лекции:

1. Сущность, понятие, задачи и функции инструментального программного обеспечения
2. Системы программирования. Трансляторы и их виды
3. Понятие языка программирования, классификация

4. Обзор языков программирования
5. Методология разработки программных продуктов

7.1.2. Краткое содержание вопросов

1. Сущность, понятие, задачи и функции инструментального программного обеспечения

Любая задача, подлежащая решению на ЭВМ, должна быть предварительно соответствующим образом подготовлена. Подготовка задачи к решению на ЭВМ включает несколько этапов. Один из основных этапов подготовки – создание алгоритма решения задачи. Алгоритм может быть представлен различными способами: для человека наиболее понятен алгоритм, изображенный в виде блок-схемы, а для ЭВМ подходит только представление алгоритма средствами алгоритмического языка программирования.

Алгоритм, записанный средствами языка программирования, называется программой. **Программа – это логически упорядоченная последовательность команд, реализующая алгоритм решения задачи.** ЭВМ понимает только язык двоичных кодов. Поэтому программа (алгоритм) для нее должна быть написана на **машинном языке** (языке низкого уровня). В то же время записать программу в машинных кодах – достаточно сложная задача, требующая соответствующего уровня подготовки программиста. В повседневной жизни при решении различных прикладных задач на ЭВМ не всегда имеется возможность привлечь специалиста по программированию, поэтому составлять программу вынужден тот, кто заинтересован в решении задачи. Для облегчения процесса программирования разработаны специальные **языки программирования**. Языки программирования – это искусственные языки. От естественных они отличаются ограниченным числом "слов", и очень строгими правилами записи команд (операторов). Тем не менее, эти языки понятны человеку (при соответствующей подготовке последнего).

С помощью языка программирования создается не готовая программа, а только ее текст, описывающий ранее разработанный алгоритм. Чтобы получить работающую программу, надо этот текст либо сначала автоматически перевести в машинный код (для этого служат программы - **компиляторы**) и затем использовать отдельно от исходного текста, либо переводить в машинный код параллельно с выполнением программы (этим занимаются программы **-интерпретаторы**).

В ходе лекции будет дано общее представление о языках программирования, пользующихся наибольшей популярностью, дана краткая характеристика этим языкам, а также изложено понятие о системе программирования.

Сущность, понятие инструментального программного обеспечения

Инструментальное программное обеспечение (ИПО) — программное обеспечение, предназначенное для использования в ходе проектирования, разработки и сопровождения программ.

Применяется инструментальное обеспечение в фазе разработки. Инструментальное программное обеспечение - это совокупность программ, используемых для помощи программистам в их работе, для помощи руководителям разработки программного обеспечения в их стремлении проконтролировать процесс разработки и получаемую продукцию. Наиболее известными представителями этой части программного обеспечения являются программы трансляторов с языков программирования, которые помогают программистам писать машинные команды. Инструментальными программами являются трансляторы с языков Фортран, Кобол, Джо-виал, Бейсик, АПЛ и Паскаль. Они облегчают процесс создания новых рабочих программ. Однако трансляторы с языков это только наиболее известная часть инструментальных программ; существует же их великое множество.

Использование вычислительных машин для помощи в создании новых программ далеко не очевидно для людей, не являющихся профессиональными программистами.

Часто же бывает так, что профессионалы рассказывают об инструментальном (фаза разработки) и системном (фаза использования) программном обеспечении на едином дыхании, предполагая, что не посвященному в тайны их мастерства известно об этой роли инструментального программного обеспечения. Так же как и в фазе использования (для прикладных программ), системное обеспечение работает и в фазе разработки, но только совместно с инструментальным обеспечением. Инструментальное ПО или системы программирования - это системы для автоматизации разработки новых программ на языке программирования.

2. Системы программирования. Трансляторы и их виды

В **состав системы программирования**, как правило, входят следующие средства:

- трансляторы с языков высокого уровня;
- средства редактирования, компоновки и загрузки программ;
- макроассемблеры (машинно-ориентированные языки);
- отладчики машинных программ.

Таковыми средствами, в первую очередь, являются:

– **текстовый редактор (Edit)**, осуществляющий функции записи и редактирования **исходного текста программы**. В принципе, для написания текста программы может быть использован любой текстовый редактор, но лучше использовать **специализированные редакторы**, которые ориентированы на конкретный язык программирования и позволяют в процессе ввода текста выделять ключевые слова и идентификаторы разными цветами и шрифтами. Подобные редакторы созданы для всех популярных языков. Как правило, они дополнительно могут автоматически проверять правильность синтаксиса программы непосредственно во время ее ввода;

– **загрузчик программ (Load)**, позволяющий выбрать из директории нужный текстовый файл программы;

– **запускающий программ (Run)**, осуществляющий процесс выполнения программы;

– **компилятор (Compile)**, предназначенный для компиляции или интерпретации исходного текста программы в машинный код с диагностикой синтаксических и семантических (логических) ошибок;

– **отладчик (Debug)**, выполняющий сервисные функции по отладке и тестированию программы;

– **диспетчер файлов (File)**, предоставляющий возможность выполнять операции с файлами: сохранение, поиск, удаление и т. п.

3. Понятие языка программирования, классификация

Сегодня практически все программы для ПЭВМ создаются с помощью **языков программирования**. Теоретически программу можно записать и средствами обычного человеческого (естественного) языка – это называется программированием на **метаязыке** (подобный подход обычно используется на этапе составления алгоритма), но автоматически перевести такую программу в машинный код пока не представляется возможным из-за высокой неоднозначности естественного языка.

Языки программирования – 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** языка программирования представляет собой фиксированный для данного языка набор основных символов, допускаемых для составления текста программы на этом языке, **синтаксис** – это система правил, определяющих допустимые конструкции языка программирования из букв алфавита, **семантика** – система правил однозначного толкования отдельных языковых конструкций, позволяющих воспроизвести процесс обработки данных.

Взаимодействие синтаксических и семантических правил определяют 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**.

Рассмотрим некоторые основные понятия, общие для большинства языков программирования.

Оператор – одно из ведущих понятий всех языков программирования. Каждый оператор представляет собой законченную фразу языка и определяет однозначно трактуемый этап обработки данных.

Большая часть операторов ведет обработку **величин**. **Величины** могут быть **постоянными** и **переменными**. Значения **постоянных величин** не изменяются в ходе выполнения программы. **Переменная** – это программный объект, способный принимать некоторое значение с помощью оператора присваивания. В ходе выполнения программы значения переменной могут неоднократно изменяться. Каждая переменная после ее описания отождествляется с некоторой ячейкой памяти, содержимое которой является ее значением.

Любая величина характеризуется **типом, именем и значением**. Наиболее распространенные типы величин – числовые (целые и вещественные), символьные, логические.

Всем программным объектам в языках программирования даются индивидуальные **имена**. Имя программного объекта называют **идентификатором**. Чаще всего идентификатором является любая конечная последовательность букв и цифр, начинающаяся с буквы.

Функция – это программный объект, задающий вычислительную процедуру определения значения, зависящего от некоторых аргументов. Функции используются в программе в **выражениях**. **Выражения** строятся из величин – постоянных и переменных, функций, скобок, знаков операций и т. д. Выражение имеет определенный тип, определяемый типом принимаемых в итоге его вычисления значений. Возможны выражения **арифметические, логические, символьные, строковые** и т. д.

Модуль – это специальная программная единица, предназначенная для создания библиотек и разделения больших программ на логически связанные блоки.

Язык программирования составляет **ядро системы программирования**. Существующие языки программирования можно разделить на две группы: **процедурные** и **непроцедурные** (рис.1).

Процедурные (алгоритмические) языки предназначены для написания процедурных программ, представляющих собой систему предписаний для решения конкретной задачи. Роль компьютера в этом случае сводится к механическому выполнению этих предписаний.

Процедурные языки программирования разделяют на языки **низкого** и **высокого** уровня.

Каждый процессор имеет **систему команд** – **совокупность машинных кодов**, которые он понимает и может исполнять. Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется **языком программирования низкого уровня** (машинно-ориентированным). Это вовсе не означает, что он "плохой". Имеется в виду, что операторы языка близки к машинному коду и ориентированы на конкретные команды процессора. Языком программирования самого низкого уровня является **язык ассемблера**. Языки программирования низкого уровня позволяют создавать программы из машинных кодов, обычно в шестнадцатеричной форме. С ними трудно работать, но созданные с их помощью высококвалифицированным программистом программы занимают меньше места в памяти и работают быстрее. С помощью этих языков удобнее разрабатывать системные

программы, драйверы, некоторые другие виды программ. Поскольку наборы инструкций для каждого типа процессора отличаются, конкретной компьютерной архитектуре соответствует свой язык программирования низкого уровня, и написанная на нем программа может быть использована только в этой среде.

Языки программирования, имитирующие естественные языки, обладающие укрупненными командами, ориентированными на решение содержательных прикладных задач, называют **языками программирования высокого уровня**. В настоящее время насчитывается несколько сотен таких языков, а если считать и их диалекты, то это число возрастет до нескольких тысяч. Языки программирования высокого уровня существенно отличаются от машинно-ориентированных 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: переслать число в ячейку; считать число из 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.

Языки программирования высокого уровня имеют, по сравнению с языками программирования низкого уровня, 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 действия 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.

Языки программирования высокого уровня значительно ближе и понятнее человеку, чем компьютеру. Особенности конкретных архитектур ПЭВМ в них не учитываются, поэтому создаваемые программы на уровне исходных текстов легко переносимы на другие платформы, для которых создан **транслятор** этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, а ошибок при создании программ допускается значительно меньше.

Языки программирования высокого уровня являются **алгоритмическими языками**. **Алгоритмический язык** – это набор символов и терминов, которые в соответствии с правилами синтаксиса описывают алгоритм решения задачи. Своими конструкциями и правилами написания этот язык, с одной стороны, близок к математическому описанию задачи, а с другой стороны – содержит такие выражения, которые близки к естественному языку, чаще всего к английскому.

Программу на алгоритмическом языке записывают, как правило, в обычном текстовом редакторе. В результате получают текстовый файл, содержащий программный код, записанный на языке программирования высокого уровня. Такой текстовый файл называют **исходным модулем (исходным текстом)**. Исходный текст программы состоит из специальных команд (**операторов языка программирования**). Процессор исполнить эти команды не может, поэтому исходный текст с помощью **транслятора (компилятора)**

или **интерпретатора**) преобразуют в инструкции процессору. Назначение компиляторов и интерпретаторов, а также принципиальные отличия между ними будут рассмотрены в третьем вопросе.

4. Обзор языков программирования

Рассмотрим основные сведения об алгоритмических языках программирования, получивших наибольшее распространение среди программистов.

Fortran (Фортран). Это первый компилируемый язык, созданный Джимом Бэкусом в 50-е годы. Программисты, разрабатывавшие программы исключительно на ассемблере, выражали серьезное сомнение в возможности появления высокопроизводительного языка высокого уровня, поэтому основным критерием при разработке компиляторов Фортрана являлась эффективность исполняемого кода. Хотя в Фортране впервые был 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 статистических комплексов и заканчивая пакетами управления спутниками, поэтому Фортран продолжает активно использоваться во многих организациях.

Cobol (Кобол). Это компилируемый язык для применения в экономической области и решения бизнес-задач, разработанный в начале 60-х годов. Он отличается большой "многословностью" – его операторы иногда 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 приложений, которые активно эксплуатируются и сегодня. Достаточно сказать, что наибольшую зарплату в США получают программисты на Коболе.

Algol (Алгол). 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 **Алгол 68**, по своим возможностям и сегодня опережающая многие языки программирования, однако из-за отсутствия достаточно эффективных компьютеров для нее не удалось своевременно создать хорошие компиляторы. Язык Алгол сыграл большую роль в теории, но для практического программирования сейчас почти не используется.

Pascal (Паскаль). Язык Паскаль, созданный в конце 70-х годов основоположником множества идей современного программирования Никлаусом Виртом и названный в честь ученого Блеза Паскаля, во многом напоминает Алгол, но в нем ужесточен ряд требований к структуре программы и имеются возможности, позволяющие успешно применять его при создании крупных программных проектов. Изначально Паскаль был задуман как язык программирования для студентов. Он и сейчас является основным языком программирования, используемым в качестве учебного, во многих высших учебных заведениях. На базе языка Паскаль созданы несколько более мощных языков (Модула, Ада, Дельфи).

BASIC (Бейсик). Для этого языка имеются и компиляторы, и интерпретаторы, а по популярности он занимает первое место в мире. Бейсик был создан в 1964 г. Томасом Куртом и Джоном Кемени как язык для начинающих программистов, облегчающий написание простых программ. В настоящее время разработано несколько различных версий языка Бейсик, таких как GWBASIC, QBASIC, Pro-BASIC, Turbo-BASIC, Visual-BASIC и др.

C (Си). 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 типа процессора.

Си во многом похож на Паскаль и имеет дополнительные средства для прямой работы с памятью (**указатели**). На этом языке в 70-е годы написано множество

прикладных и системных программ и ряд известных операционных систем (например, Unix). В настоящее время Си широко используется при создании системного программного обеспечения.

PL/1 (ПЛ/1). В середине 60-х годов компания **IBM** решила взять все лучшее из языков Фортран, Кобол и Алгол и воплотить в одном языке программирования. В результате в 1964 году на свет появился новый компилируемый язык программирования, который получил название **Programming Language One**. В этом языке было реализовано множество уникальных решений, полезность которых удастся оценить только спустя 33 года, в эпоху крупных программных систем. По своим возможностям **ПЛ/1** значительно мощнее многих других языков (Си, Паскаля). Например, в **ПЛ/1** присутствует уникальная возможность указания точности вычислений – ее нет даже у **Си++** и **Явы**. В настоящее время язык **ПЛ/1** почти не используется.

Ada (Ада). Назван по имени леди Огасты Ады Лавлейс, дочери английского поэта Байрона и его отдаленной родственницы Анабеллы Милбэнк. В 1979 году сотни экспертов Министерства обороны США отобрали из 17 вариантов именно этот язык, разработанный небольшой группой под руководством Жана Ихбиа. Он удовлетворил на то время все требования Пентагона, а к сегодняшнему дню в его развитие вложены десятки миллиардов долларов. Структура самого языка похожа на Паскаль. В нем имеются средства строгого разграничения доступа к различным уровням спецификаций, доведена до предела мощность управляющих конструкций.

Принципиально иное направление в программировании связано с методологиями **непроцедурного программирования**. Для этих целей используются **непроцедурные языки программирования**, которые можно разделить на **объектные** и **декларативные**. Объектно-ориентированный язык создает окружение в виде **множества независимых объектов**. Каждый объект ведет себя подобно отдельному компьютеру, их можно использовать для решения задач как "черные ящики", не вникая во внутренние механизмы их функционирования. Объектно-ориентированный подход к проектированию программных продуктов основан на:

- выделении классов объектов;
- установлении характерных **свойств** объектов и **методов** их обработки;
- создании иерархии классов, наследовании свойств объектов и методов их обработки.

Каждый объект объединяет как **данные**, так и **программу обработки** этих данных и относится к определенному классу. С помощью класса один и тот же программный код можно использовать для относящихся к нему различных объектов.

Из языков объектного программирования, популярных среди профессионалов, следует назвать прежде всего:

С++ (Си++). **Си++** - это объектно-ориентированное расширение языка Си, созданное Бьярном Страуструпом в 1980 году. Множество новых мощных возможностей, позволивших резко повысить производительность работы 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 (Джава). Этот язык был создан компанией Sun в начале 90-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 платформенно-независимый байт-код (каждая команда занимает один байт). Этот байт-код может выполняться с помощью интерпретатора – виртуальной Java-машины, версии которой созданы сегодня для любых платформ. Благодаря наличию множества Java-машин программы на Джава можно переносить не только на уровне исходных текстов, но и на уровне двоичного байт-кода, поэтому по популярности язык Джава сегодня занимает второе место в мире после Бейсика. Язык

Джава чрезвычайно эффективен для создания интерактивных Web-страниц;

Smalltalk (Смолток). Работа над этим языком началась в 1970 году в исследовательской лаборатории корпорации XEROX, а закончилась спустя 10 лет, воплотившись в окончательном варианте интерпретатора **SMALLTALK-80**. Данный язык оригинален тем, что его синтаксис очень компактен и базируется исключительно на понятии объекта. В этом языке отсутствуют операторы или данные. Все, что входит в Смолток, является объектами, а сами объекты общаются друг с другом исключительно с помощью сообщений (например, появление выражения **I+1** вызывает посылку объекту **I** сообщения "+", то есть "прибавить", с параметром 1, который считается не числом-константой, а тоже объектом). Больше никаких управляющих структур, за исключением "оператора" ветвления (на самом деле функции, принадлежащей стандартному объекту), в языке нет, хотя их можно очень просто смоделировать. Сегодня версия **VisualAge for Smalltalk** активно развивается компанией IBM;

Delphi (Дельфи) – язык объектно-ориентированного "визуального" программирования. Явился развитием языка Паскаль.

При использовании **декларативного** языка программирования программист указывает исходные информационные структуры, взаимосвязи между ними и то, какими свойствами должен обладать результат. При этом процедуру его получения (то есть алгоритм) программист не строит. В этих языках отсутствует понятие "оператор".

Типичными представителями декларативных языков являются:

LISP (Лисп). Интерпретируемый язык программирования, созданный в 1960 году Джоном Маккарти. Ориентирован на структуру данных в форме списка и позволяет организовывать эффективную обработку больших объемов текстовой информации;

Prolog (Пролог). Создан в начале 70-х годов. Программа на этом языке, в основу которого положена математическая модель теории исчисления предикатов, строится из последовательности фактов и правил, а затем формулируется утверждение, которое Пролог будет пытаться доказать с помощью введенных правил. Человек только описывает структуру задачи, а внутренний "мотор" Пролога сам ищет решение с помощью методов поиска и сопоставления.

5. Методология разработки программных продуктов

Классификация методов проектирования программных продуктов

Проектирование алгоритмов и программ - наиболее ответственный этап жизненного цикла программных продуктов, определяющий, насколько создаваемая программа соответствует спецификациям и требованиям со стороны конечных пользователей. Затраты на создание, сопровождение и эксплуатацию программных продуктов, научно-технический уровень разработки, время морального устаревания и многое другое- все это также зависит от проектных решений.

Пример: Переход к графической среде работы конечного пользователя типа Windows или Macintosh потребует создания пользовательского интерфейса с элементами управления в виде пиктограмм, кнопок, выпадающих меню, обязательного применения манипулятора мышью и др. Отсутствие в программном продукте уже ставших стандартом подобных элементов свидетельствует о том, что в будущем потребуются значительные затраты на модификацию этого продукта, иначе будет падать его конкурентоспособность и привлекательность для конечного пользователя.

8.1. Лекция № 10, 11 (4 часа)

Тема: Программирование на алгоритмическом языке QBASIC

8.1.1. Вопросы лекции:

1. Основные понятия алгоритмического языка QBASIC. История развития QBASIC.

Операции над числами и данными

2. Встроенные функции

- 3. Арифметические выражения
- 4. Основные операторы алгоритмического языка QBASIC
 - 4.1. Оператор присваивания LET
 - 4.2. Операторы ввода данных READ, DATA, RESTORE, INPUT
 - 4.3. Операторы вывода данных PRINT, LPRINT
 - 4.4. Условные операторы IF ... THEN, GOTO, SELECT CASE
 - 4.5. Операторы организации цикла FOR ... NEXT, WHILE ... WEND, DO ... LOOP
 - 4.6. Операторы: REM, CLS, STOP, END

8.1.2. Краткое содержание вопросов

1. Основные понятия алгоритмического языка QBASIC. История развития QBASIC. Операции над числами и данными

Бейсик (от BASIC, сокращение от англ. Beginner's All-purpose Symbolic Instruction Code — универсальный код символических инструкций для начинающих; англ. basic — основной, базовый) — семейство высокоуровневых языков программирования.

Был разработан в 1963 году профессорами Дартмутского колледжа Томасом Курцем и Джоном Кемени.

Язык предназначался для обучения программированию и получил широкое распространение в виде различных диалектов, прежде всего, как язык для домашних компьютеров.

В 1975 году, Пол Аллен, молодой программист из Бостона, в содружестве со студентом Гарвардского университета Биллом Гейтсом написали программу, реализующую для микрокомпьютера Алтаир 8800 язык Бейсик, впервые использовав его для программного обеспечения персональных компьютеров. Впоследствии Гейтс и Аллен основали собственную фирму Microsoft.[1]

Алгоритмический язык Basic используется преимущественно в режиме диалога человека и ЭВМ. Этот язык ориентирован на решение различных задач вычислительного и невычислительного характера с небольшим объемом исходной информации. Название языка BASIC возникло от сокращения английских слов Beginner's All-purpose Symbolic Instruction Code (многоцелевой язык символических инструкций для начинающих). Следует отметить, что стандарта на язык Basic не существует и различные его модификации могут существенно отличаться друг от друга.

В языке Basic существуют как средства для описания действий алгоритма, которые используются при составлении программ, — операторы Basic, так и средства, которые служат для общения с ЭВМ. Последние имеют форму приказов для немедленного выполнения и называются командами.

Основным режимом в Basic является программный режим, когда заранее составленная программа полностью вводится в ЭВМ и затем выполняется.

Программа на Basic состоит из строк, которые могут иметь номер. В одной строке может содержаться один или несколько операторов, разделенных символом : (двоеточие). Обычно строки нумеруются, начиная с 10, с шагом 10. Номера строк используются в операторах передачи управления. При этом оператор, которому передается управление, должен быть первым в строке.

2. Встроенные функции

Для вычисления наиболее распространенных элементарных математических функций в языке БЕЙСИК применяют встроенные стандартные функции. Аргумент функции заключается в круглые скобки. Аргументом функции может быть произвольное арифметическое выражение. Для обращения к функции нужно набрать ее имя и указать аргумент. Стандартные математические функции, применяемые в языке Бейсик, приведены в ТАБЛИЦЕ 1.

Аргументы тригонометрических функций должны быть заданы в радианах. Если угол выражен в градусах, то его нужно перевести в радианы по формуле

Значение в радианах = значение в градусах * 3.14/180

В некоторых версиях языка Бейсик имеется функция LOG(X) для вычисления натурального логарифма числа X. Однако от логарифма с любым основанием легко перейти к натуральному логарифму, используя формулу

$\text{Log}_a N = \ln N / \ln a$, где a – основание.

Над числовыми константами, переменными и стандартными функциями можно производить обычные арифметические операции:

1. вычисляются значения функций
2. ^ возведение в степень,
3. * умножение, / деление,
4. + сложение, - вычитание.

ТАБЛИЦА 1

Обозначение функции на языке Бейсик	Название	Пояснение
ABS(X)	Функция абсолютная величина x (модуль)	Функция вычисляет абсолютное значение (модуль) аргумента X
SQR(X)	Функция квадратный корень ($\sqrt{x}$)	Функция вычисляет квадратный корень положительного аргумента X
LOG(X)	Функция натуральный логарифм $\ln x$	Функция вычисляет натуральный логарифм положительного аргумента X
EXP(X)	Экспоненциальная функция e^x	Вычисляет e^x , где $e=2,71828$, X-любое число
COS(X)	Функция косинуса $\cos x$	Вычисляет косинус аргумента X
SIN(X)	Функция синуса $\sin x$	Вычисляет синус аргумента X
TAN(X)	Функция тангенса $\tan x$	Вычисляет тангенс аргумента X, находящегося в диапазоне от $-\frac{\pi}{2}$ до $\frac{\pi}{2}$
ATN(X)	Функция арктангенса $\arctg x$	Вычисляет арктангенс аргумента X
INT(X)	Целочисленная функция	Определяет наибольшее целое число, не превосходящее X
RND(X)	Функция случайных чисел	Выдает случайное число, лежащее в интервале от 0 до 1. Значение аргумента X игнорируется

3. Арифметические выражения

Арифметическое выражение – это символическая запись, указывающая правила вычисления числового выражения. Оно состоит из чисел (констант), имен переменных, функций, знаков арифметических операций и скобок.

При записи выражений необходимо учитывать следующие рекомендации и ограничения:

1. Формулу требуется записывать в строку без каких-либо подстрочных и надстрочных знаков.

2. Следует использовать круглые скобки для указания порядка действий, особенно в сомнительных случаях. Вычисления в скобках производятся в первую очередь. Если выражение, содержащее скобки само заключено в скобки, то вычисления производятся, начиная с внутренних скобок. Внутри скобок действия выполняются с лева на право в соответствии с приоритетом операций:

- Сначала вычисляются значения функций,
- Затем - выполняются все операции возведения в степень,
- Затем - умножения и деления,
- И наконец, сложения и вычитания.

Если математическое выражение содержит несколько операций одинакового старшинства, то такие операции выполняются последовательно слева на право.

3. Два знака арифметических операций нельзя ставить рядом, а также нельзя опускать знак умножения между сомножителями.

4. Выполнение арифметических операций над арифметическими выражениями одного типа дает результат того же типа. Операция над целой и вещественной величиной дает вещественный результат.

5. Возведение в целую степень выполняется многократным умножением или с помощью операции $\wedge$ возведения в степень.

6. Вычисление результата возведения в степень осуществляется с помощью функций EXP и LOG, если показатель степени - действительное число:

$$x^y = \text{EXP}(Y * \text{LOG}(X))$$

Так как логарифмы отрицательных чисел не существуют, то нельзя возводить отрицательные числа в действительную степень.

3.4. Основные операторы алгоритмического языка QBASIC

В результате изучения студент должен знать:

- Понятие оператора;
- Оператор присваивания в Бейсике;
- Операторы задания значений переменным в Бейсике;
- Оператор вывода в Бейсике;
- Оператор ввода в Бейсике;
- Оператор комментариев в Бейсике;
- Оператор очищения экрана;
- Оператор конца текста программы;
- Оператор безусловного перехода;
- Оператор условного перехода в полной форме;
- Оператор условного перехода в неполной форме.

уметь:

- Приводить примеры использования операторов языка программирования Бейсик;
- Использовать операторы языка программирования Бейсик при составлении простейших программ.

Оператор – это предписание ЭВМ, написанное на языке БЕЙСИК. Операторы можно разделить на две группы:

1. Выполняемые операторы – определяют действие программы, указывая Бейсик-системе, какую операцию нужно выполнить (PRINT – печатать, INPUT - ввести);

2. Невыполняемые операторы – описывают характер и упорядочение данных, позволяют вводить в программу примечания и сообщения описательного характера (DATA, REM).

9.1. Лекция № 12, 13 (4 часа)

Тема: Массивы. Обработка одномерных массивов

9.1.1. Вопросы лекции:

1. Понятие массивов. Оператор DIM
2. Стандартные алгоритмы обработки элементов одномерных массивов
 - 2.1. ввода-вывода элементов одномерного массива
 - 2.2. расчета суммы (произведения)
 - 2.3. организации различных счетчиков
 - 2.4. выбора максимального (минимального) элемента массива
 - 2.5. упорядочивания элементов массива по возрастанию (убыванию) их значений

9.1.2. Краткое содержание вопросов

1. Понятие массивов. Оператор DIM

Многие задачи, которые решаются с помощью ЭВМ, связаны с обработкой больших объемов информации, представляющей совокупность данных, объединенных единым математическим содержанием или связанных между собой по смыслу. Примером таких организованных совокупностей данных являются координаты, задающие положение точки в пространстве, матрица коэффициентов, определяющая систему линейных уравнений, значения некоторой функции в произвольных точках, коэффициенты многочлена и т. д. Такие данные удобно представлять в виде линейных или прямоугольных таблиц.

В линейной таблице каждому ее элементу соответствует порядковый номер. Для элемента прямоугольной таблицы должны быть указаны два номера: номер по вертикали (номер строки) и номер по горизонтали (номер столбца). В высшей математике табличные величины называют соответственно векторами и матрицами.

В программе для представления таких данных используются массивы.

Массив — это упорядоченная совокупность однотипных данных, с каждым из которых связан упорядоченный набор целых чисел, называемых индексами. Массив характеризуется именем, типом и размером.

Имя массива образуется по общему правилу образования имен, т. е. представляет собой идентификатор, например A, B1 C8 и т. д. Однако оно не должно совпадать с именем ни одной простой переменной, используемой в той же программе.

Переменная с индексами

Работа с массивом сводится к действиям над его элементами. Для того чтобы указать, какой элемент в данный момент используется, достаточно задать его порядковый номер, который приписывается к имени соответствующего массива. Таким образом, элементы массива обозначаются переменной с индексами. Запись переменной с индексами состоит из имени массива и следующего за ним в круглых скобках списка индексов, например A(1), A(I), B1(K), C8(I, J), C8(2, 1)

В качестве индексов можно использовать константы, переменные, а также арифметические выражения, называемые индексными выражениями, например A1(K+5), Q3(3*A+B, C(I)-2). Если в качестве индекса используется выражение, то перед обращением к элементу массива (его записью в программе) осуществляется вычисление индексного выражения и выделение целой части. Естественно, что все переменные, входящие в выражение, а также переменные, используемые в качестве индексов, должны быть заданы до обращения. Например, запись T(2*Y-3.2, Z+1.7) при $y = 1,8$ и $2 = 0,9$ соответствует элементу T(0,2).

Индексы определяют положение элемента в массиве. Число индексов определяет размерность массива, т. е. форму его компоновки: одномерный, двумерный и т. д. Одномерный массив соответствует линейной таблице. Его элемент обозначается

переменной с одним индексом: $A(1)$, $A(1)$ - соответственно первый и 2-й элементы одномерного массива A ; $B(K)$ — k -й элемент одномерного массива B . Двумерный массив описывает в программе прямоугольную таблицу. Его элементы обозначаются переменной с двумя индексами: $C8(I, J)$, $C8(2, 1)$, где первый индекс обозначает номер строки, а второй — номер столбца. В языке Бейсик допускается работа с одномерными и двумерными массивами.

Таким образом, для обращения к конкретному элементу массива необходимо указать имя массива и значения индексов. Напомним, что если в качестве индексов используются переменные или выражения, то значения всех указанных переменных должны быть определены до обращения.

Описание массива

Для записи элементов массива в память ЭВМ нужно выделить для их хранения необходимое количество (массив) ячеек памяти, которое определяется размером массива. Размеры массива задаются границами изменения индексов по каждому измерению. Минимальное значение индексов равно нулю. С учетом этого количество элементов в одномерном массиве будет равно значению верхней границы индекса плюс единица, а двумерного — произведению увеличенных на единицу значений верхних границ индексов.

В программе для каждого массива должны быть указаны его параметры: имя, размерность и размеры. Эта информация нужна для резервирования необходимого объема памяти для хранения числовых значений; она задается специальным оператором описания массивов DIM (сокращение от DIMENSION — размерность). Оператор DIM в общем случае имеет следующий вид:

$\langle n \rangle \text{ DIM } V1(i_1[j_1]), V2(i_2[j_2]), \dots, V_k(i_k[j_k])$

где n — номер строки; DIM—имя оператора; $V1, V2, \dots, V_k$ — имена массивов; $i_1, j_1, i_2, j_2, \dots, i_k, j_k$ — верхние границы индексов, представляющие собой положительные целые константы; $[, j_k]$ — квадратные скобки здесь указывают на необязательность элемента. Например, если в программе используется одномерный массив A , содержащий 10 элементов, и двумерный массив $B1$, состоящий из трех строк и пяти столбцов, то оператор описания массивов будет иметь вид

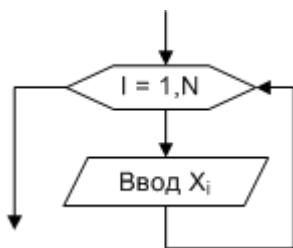
10 DIM A(9), B1(2, 4)

Оператор DIM может быть записан в любом месте программы до первого оператора, использующего соответствующую индексную переменную. Однако наиболее употребительной является такая структура, при которой все описания объединяются в начале программы. Выполняя оператор DIM, машина выделяет для массива соответствующее количество ячеек памяти, причем имя массива определяет адрес первой ячейки участка памяти, хранящего данный массив. Элементы массивов располагаются в памяти ЭВМ один за другим в последовательных ячейках памяти. Строгая последовательность записи элементов исключает необходимость хранения порядковых номеров элементов.

2. Стандартные алгоритмы обработки элементов одномерных массивов

2.1. ввода-вывода элементов одномерного массива

При вводе массива необходимо последовательно вводить 1-й, 2-й, 3-й и т.д. элементы массива, аналогичным образом поступить и при выводе. Следовательно, необходимо организовать цикл.



Как видно, безусловный цикл удобно использовать для обработки всего массива, и в дальнейшем при выполнении таких операций будем применять именно его. Вывод массива организуется аналогично вводу.

Рассмотрим несколько примеров обработки массивов. Алгоритмы, с помощью которых обрабатывают одномерные массивы, похожи на обработку последовательностей (вычисление суммы, произведения, поиск элементов по определенному признаку, выборки и т. д.). Отличие заключается в том, что в массиве одновременно доступны все его компоненты, поэтому становится возможной, например, сортировка его элементов и другие, более сложные преобразования.

2.2. расчета суммы (произведения)

Дан массив X , состоящий из n элементов. Найти сумму элементов этого массива. Процесс накапливания суммы элементов массива достаточно прост и практически ничем не отличается от суммирования значений некоторой числовой последовательности. Переменной S присваивается значение равное нулю, затем последовательно суммируются элементы массива X . Блок-схема алгоритма расчета суммы приведена на рис. 12.

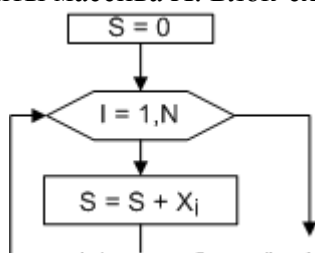


Рис. 12. Алгоритм вычисления суммы элементов массива

Дан массив X , состоящий из n элементов. Найти произведение элементов этого массива. Решение этой задачи сводится к тому, что значение переменной P , в которую предварительно была записана единица, последовательно умножается на значение i -го элемента массива. Блок-схема алгоритма приведена на рис. 13.

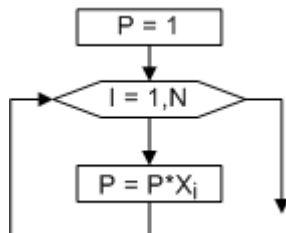


Рис. 13. Вычисление произведения элементов массива

2.3. организации различных счетчиков

2.6. удаления (добавления) элемента массива

Сортировка вставкой

Сортировка вставкой заключается в том, что сначала упорядочиваются два элемента массива. Затем делается вставка третьего элемента в соответствующее место по отношению к первым двум элементам. Четвертый элемент помещают в список из уже

упорядоченных трех элементов. Этот процесс повторяется до тех пор, пока все элементы не будут упорядочены.

Прежде чем приступить к составлению блок-схемы рассмотрим следующий пример. Пусть известно, что в массиве из восьми элементов первые шесть уже упорядочены, а седьмой элемент нужно вставить между вторым и четвертым. Сохраним седьмой элемент во вспомогательной переменной, так как показано на рисунке 17, а на его место запишем шестой. Далее пятый переместим на место шестого, четвертый на место пятого, а третий на место четвертого, тем самым, выполнив сдвиг элементов массива на одну позицию вправо. Записав содержимое вспомогательной переменной в третью позицию, достигнем нужного результата.

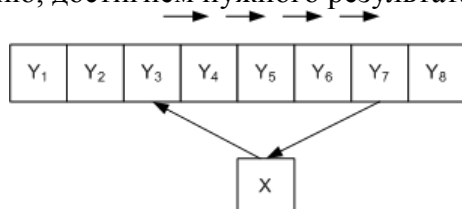


Рис. 17. Процесс вставки элемента в массив

Составим блок-схему алгоритма (рис. 18), учитывая, что возможно описанные выше действия придется выполнять неоднократно.

Организуем цикл для просмотра всех элементов массива, начиная со второго (блок 4). Сохраним значение текущего i -го элемента во вспомогательной переменной X , так как оно может быть потеряно при сдвиге элементов (блок 5) и присвоим переменной j значение индекса предыдущего $(i-1)$ -го элемента массива (блок 6). Далее движемся по массиву влево в поисках элемента меньшего, чем текущий и пока он не найден сдвигаем элементы вправо на одну позицию. Для этого организуем цикл (блок 7), который прекратиться, как только будет найден элемент меньше текущего. Если такого элемента в массиве не найдется и переменная j станет равной нулю, то это будет означать, что достигнута левая граница массива, и текущий элемент необходимо установить в первую позицию. Смещение элементов массива вправо на одну позицию выполняется в блоке 8, а изменение счетчика j в блоке 9. Блок 10 выполняет вставку текущего элемента в соответствующую позицию.

10.1. Лекция № 14 (2 часа)

Тема: Программные средства реализации информационных процессов. Сервисное программное обеспечение

10.1.1. Вопросы лекции:

1. Назначение и виды сервисных программ
2. Программы контроля и диагностики компьютера. Файловые менеджеры
3. Программы записи, обслуживания магнитных и компакт-дисков
4. Программы обслуживания ОС Windows

10.1.2. Краткое содержание вопросов

1. Назначение и виды сервисных программ.

В процессе практической работы на компьютере у пользователя периодически возникают проблемы, связанные как с работой системы в целом, так и отдельных устройств и программ: замедление работы, нехватка места на диске, ошибки выполнения программ, их зависание и т.п. Для выявления таких проблем, устранения причин их возникновения и обслуживания отдельных устройств в состав системного программного обеспечения включают сервисные программы (утилиты).

Сервисное программное обеспечение – это совокупность программ, которые предназначены для тестирования устройств компьютера и обеспечения нормальной работы основных приложений.

Существует два типа сервисных программ – стандартные, входящие в состав операционной системы, и специализированные, которые часто объединяют в комплексы, подобные Norton Utilities или Acronis Power Utilities. Специализированные программы не только дублируют функции системных утилит в более удобном режиме, но и значительно расширяют их возможности по обслуживанию устройств компьютера, хранению и защите данных.

Перечень сервисных программ, представленных на рынке ПО, очень широк; многие из них дублируют друг друга. По функциональному назначению их можно разделить на несколько групп:

- программы контроля и диагностики компьютера;
- файловые менеджеры;
- антивирусные программы;
- программы обслуживания дисков;
- программы работы с архивами;
- программы обслуживания операционной системы;
- программы обслуживания сети.

2. Программы контроля и диагностики компьютера. Файловые менеджеры

Программы контроля и диагностики компьютера

Данный класс программ предназначен для проверки наличия и работоспособности устройств, определения их характеристик и технических параметров работы.

Существует три уровня тестирования устройств.

Первый уровень – самотестирование базовой системой ввода-вывода (BIOS) при включении компьютера (POST или Power-On-Self-Test). На этом уровне проверяется наличие и работоспособность основных компонентов компьютера: процессора, оперативной памяти, вспомогательных микросхем, приводов дисков, клавиатуры и видеоадаптера. Если тест успешно пройден, выдается один короткий звуковой сигнал и начинается загрузка ОС. Если же BIOS не находит основное устройство или возникает ошибка ввода-вывода, выдается сообщение на экран компьютера (например: «Keyboard error – ошибка клавиатуры») или последовательность звуковых сигналов, означающих определенную ошибку. Описание этих звуковых сигналов содержится в инструкции по эксплуатации материнской платы.

Второй уровень – тестирование операционной системой. ОС при загрузке определяет все подключенные устройства, наличие конфликтов между ними, соответствие им драйверов. При обнаружении ошибки выдается сообщение на экран монитора (например: «Обнаружено новое устройство» или «Ошибка принтера»), возможные причины и варианты их устранения.

Третий уровень – пользовательское тестирование, основанное на применении более основательных общих или специализированных тестов. Существуют программы для измерения производительности дисков, контроля температуры процессора, проверки качества изображения на мониторе и т.д. Имеется специализированное ПО для проверки работы различных устройств (например, *3DMark* – для тестирования видеосистемы) или тестовые программные комплексы (например, *SiSoftware Sandra*, *PCMark*).

В Windows имеется стандартная служебная программа *Сведения о системе*, которая собирает и отображает данные о ее параметрах. Сюда входит, в частности, информация о конфигурации оборудования, компонентах компьютера, установленном программном обеспечении.

Файловые менеджеры

Файловые менеджеры – это сервисные программы, предназначенные для управления файлами и папками (каталогами) на дисках. Основными операциями управления являются: просмотр содержимого папок, навигация по файловой системе (переход из одной папки в другую), поиск, копирование, перемещение и удаление файлов и папок, запуск программ и т.д. Помимо перечисленных, большинство современных файловых менеджеров предоставляют дополнительные возможности: групповые операции с файлами, упорядочение (сортировка) объектов, возможность работы с файловыми архивами как с обычными папками, средства сравнения файлов и папок и многие другие.

3. Программы записи, обслуживания магнитных и компакт-диск

Программы обслуживания магнитных дисков

Наиболее часто используемыми операциями обслуживания магнитных дисков являются их проверка и дефрагментация.

Программы проверки магнитных дисков. Для снижения риска потери информации, хранящейся на жестких и гибких дисках, рекомендуется периодически проверять их на наличие логических и физических ошибок. Физические дефекты возникают из-за механических повреждений, воздействия электромагнитных полей или старения магнитного покрытия дисков. Наличие таких дефектов на магнитной поверхности (секторах и кластерах) делает ее непригодной для хранения информации. Если в этой области записана информация, то полностью ее прочитать будет невозможно. Дефектные сектора и кластеры диска должны быть исключены из дальнейшего использования программой проверки.

4. Программы обслуживания ОС Windows

Программы оптимизации работы ОС позволяют повысить быстродействие компьютера за счет корректной установки и удаления программного обеспечения, более эффективного распределения оперативной памяти, оптимизации файлов реестра, отключения ненужных для использования компонент и служб операционной системы.

Чтобы программа правильно и без сбоев работала в ОС Windows, она должна быть корректно установлена на компьютер.

Установка программного обеспечения – это процедура добавления файлов и папок на жесткий диск и соответствующих данных в системный реестр Windows. После установки в меню и на рабочем столе появляются необходимые для запуска программы ярлыки, а сами программы настраиваются на работу с устройствами компьютера.

Программы устанавливаются с *дистрибутивных дисков* (как правило, это CD), на которых находится программа-установщик (инсталлятор) с совокупностью исходных файлов и папок (обычно в сжатом виде), а также разного рода дополнительная информация. Чаще всего инсталлятор носит имя *setup.exe* (или *install.exe*); для установки программы нужно запустить его на исполнение и следовать выдаваемым на экран инструкциям.

Неиспользуемые или неправильно работающие программы необходимо удалять из компьютера. *Удаление программного обеспечения (Uninstall, деинсталляция)* – это процедура удаления файлов и папок программы с жесткого диска и соответствующих данных из системного реестра Windows. К сожалению, стандартные средства Windows далеко не всегда осуществляют эту операцию корректно; зачастую в системных файлах остаются записи о давно удаленных программах. Со временем накапливающийся в конфигурационных файлах «мусор» начинает утяжелять систему, делая ее менее устойчивой и менее производительной.

Файлы реестра Windows представляют собой специализированную базу данных, в которой хранится информация об аппаратной и программной конфигурации компьютера. Программы оптимизации работы ОС (*RegCleaner, Customizer XP* и др.) проводят анализ

правильности хранения данных в реестре, удаляют ссылки на отсутствующие программы и упорядочивают структуру данных, что приводит к определенному повышению быстродействия системы.

Некоторые программы (например, *Ashampoo Uninstaller*) специально предназначены для мониторинга изменений, происходящих при установке программ, что облегчает в дальнейшем их корректное удаление из системы.

Существует ряд интегрированных программ (например, *WinXP Manager*), позволяющих осуществлять полную оптимизацию работы ОС: повышение скорости загрузки и выключения, очистку жесткого диска от ненужных файлов и реестра от ненужных ссылок, а также тонкую настройку параметров операционной системы (отключение неиспользуемых служб, ненужных компонентов и т.д.).

Для многих современных программ, в том числе для операционных систем, характерна большая сложность настройки; рядовому пользователю может потребоваться очень много времени, чтобы во всем этом разобраться. Поэтому появились программы тонкой настройки (*Tweak*), которые дают возможность пользователю в удобном и понятном для него диалоговом режиме изменять важные параметры работы ОС. Программы тонкой настройки существуют и для других видов ПО.

11.1. Лекция № 15 (2 часа)

Тема: Программные средства реализации информационных процессов. Прикладное программное обеспечение

11.1.1. Вопросы лекции:

1. Классификация прикладного программного обеспечения
2. Понятие ППП
3. Структура и основные компоненты ППП
4. Этапы развития ППП

11.1.2.. Краткое содержание вопросов

1. Классификация прикладного программного обеспечения

Общее понятие ПО для ПК.

Персональный компьютер, как известно, является универсальным устройством для обработки информации. Персональные компьютеры могут выполнять любые действия по обработке информации. Для этого необходимо составить для компьютера на понятном ему языке точную и подробную последовательность инструкций – программу, как надо обрабатывать информацию.

Меняя программы для компьютера, можно превратить его в рабочее место бухгалтера или конструктора, дизайнера или ученого, писателя или агронома. Кроме того, тенденция понижения стоимости компьютерной техники при одновременном росте ее производительности привела к тому, что компьютеры становятся предметом домашнего обихода, как, например, телевизор или холодильник, что расширяет сферу применения ПК еще больше. Соответственно, требуется все более разнообразное программное обеспечение для решения задач в новых областях применения ПК. Непрерывное повышение мощности персональных компьютеров, периферийных устройств, а также развитие средств связи дает разработчикам программного обеспечения все больше возможностей для максимально полного удовлетворения запросов конечных потребителей. Это и ставший стандартом графический интерфейс для любого ПО, и внедренные возможности для отправки документов и данных с помощью Интернет непосредственно из прикладной программы (*Microsoft Word*, *Excel*, *Access* и др.), и возможность использования компьютера как хранилища информации благодаря появлению новых видов накопителей большой емкости и малым временем доступа к данным, а также многие другие возможности и сервисные функции.

2. Понятие ППП

Многочисленные программные средства для решения различных типов вычислительных задач можно разделить на 4 группы:

- отдельные прикладные программы;
- библиотеки прикладных программ;
- пакеты прикладных программ;
- интегрированные программные системы.

3. Структура и основные компоненты ППП

Несмотря на большое разнообразие конкретных пакетных разработок, можно выделить следующие основные компоненты ППП:

- входные языки;
- предметное обеспечение;
- системное обеспечение.

Важно отметить, что такое разбиение на составные элементы отражает в первую очередь функции, выполняемые программами ППП, а не структуру самих программ, которая зависит от индивидуальных особенностей конкретного пакета. В разных пакетах указанные компоненты могут быть развиты в различной степени или вовсе отсутствовать. Однако наиболее развитые ППП, как правило, обладают всеми этими компонентами, каждый из которых может иметь довольно сложную структуру. В многочисленных работах, посвященных пакетной проблематике, из-за не устоявшейся терминологии нередко используются другие названия составных элементов ППП. Например, входной язык называют также языком заданий или языком управления. Для обозначения предметного обеспечения применяются термины "функциональное наполнение", "функциональная подсистема" или "тело пакета". Системное обеспечение часто называют системным наполнением, организующей или управляющей программой, а также процессором пакета.

4. Этапы развития ППП

Пакетная проблематика в качестве самостоятельного научно направления сложилась в основном за последние 15-20 лет. Первые ППП представляли собой простые тематические подборки программ для решения отдельных задач в той или иной прикладной области. Современный пакет является сложной программной системой, включающей специализированные системные и языковые средства. В относительно короткой истории развития вычислительных ППП можно выделить 4 основных поколения (класса) пакетов. Каждый из этих классов характеризуется определенными особенностями входящих состав ППП компонентов - входных языков, предметного и системного обеспечения.

В качестве входных языков ППП первого поколения использовались универсальные языки программирования (Фортран, Алгол-60 и т. п.) или языки управления заданиями соответствующих операционных систем. Проблемная ориентация входных языков достигалась за счет соответствующей мнемоники в именах переменных, функций процедур, а также в текстовых константах. Составление заданий на таком языке практически не отличалось от написания программ на алгоритмическом языке.

Предметное обеспечение первых ППП, как правило, было организовано в форме библиотек программ, т.е. в виде наборов (пакетов) независимых программ на некотором базовом языке программирования (отсюда впервые возник и сам термин "пакет"). Такие ППП иногда называют пакетами библиотечного типа, или пакетами простой структуры.

В качестве системного обеспечения пакетов первого поколения обычно использовались штатные компоненты программного обеспечения ЭВМ: компиляторы с алгоритмических языков, редакторы текстов, средства организации библиотек программ, архивные системы и т. д. Эти пакеты не требовали сколько-нибудь развитой системной

поддержки, и для их функционирования вполне хватало указанных системных средств общего назначения. В большинстве случаев разработчиками таких пакетов были прикладные программисты, которые пытались приспособить универсальные языки программирования к своим нуждам.

Разработка ППП второго поколения осуществлялась уже с участием системных программистов. Это привело к появлению специализированных входных языков (их называют встроенными языками) на базе универсальных языков программирования. Проблемная ориентация таких языков достигалась не только за счет использования определенной мнемоники, но также применением соответствующих языковых конструкций, которые упрощали формулировку задачи и делали ее более наглядной. Транслятор с такого языка представлял собой препроцессор (чаще всего макропроцессор) к транслятору соответствующего алгоритмического языка.

В качестве модулей в пакетах этого класса стали использоваться не только программные единицы (т.е. законченные программы на том или ином языке программирования), но и такие объекты, последовательность операторов языка программирования, совокупность данных, схема счета и др.

Существенные изменения претерпели также принципы организации системного обеспечения ППП. В достаточно развитых пакетах второго поколения уже можно выделить элементы системного обеспечения, характерные для современных пакетов: монитор, трансляторы с входных языков, специализированные банки данных, средства описания модели предметной области и планирования вычислений и др.

Третий этап развития ППП характеризуется появлением самостоятельных входных языков, ориентированных на пользователей-непрограммистов. Особое внимание в таких ППП уделяется системным компонентам обеспечивающим простоту и удобство. Это достигается главным образом за счет такой специализации входных языков и включения в состав пакета средств автоматизированного планирования вычислений.

Наконец, четвертый этап характеризуется созданием ППП, эксплуатируемых в диалоговом режиме работы. Основным преимуществом диалогового взаимодействия с ЭВМ является возможность активной обратной связи с пользователем в процессе постановки задачи, ее решения и анализа полученных результатов. Появление и интенсивное развитие различных форм диалогового общения обусловлено прежде всего прогрессом в области технических средств обеспечения диалога. Сюда относится создание разнообразной дисплейной техники (растровые дисплеи, средства реализации графических, цветовых и звуковых возможностей, различные технические устройства для ведения диалога и т. д.), а также надежных и скоростных линий связи. Развитие аппаратного обеспечения повлекло за собой создание разнообразных программных средств поддержки диалогового режима работы (диалоговые операционные системы, диалоговые пакеты программ различного назначения и т. д.). Во многих приложениях диалог уже полностью заменил пакетную обработку, а построчный режим диалога уступает место поэкранному режиму и многооконному графическому способу общения.

12.1. Лекция № 16, 17 (4 часа)

Тема: Компьютерные сети

12.1.1. Вопросы лекции:

1. Основные понятия. Классификация компьютерных сетей
2. Локальные вычислительные сети
3. Глобальная вычислительная сеть
4. Требования, предъявляемые к современным вычислительным сетям

12.1.2. Краткое содержание вопросов

1. Основные понятия. Классификация компьютерных сетей

По мере того, как в информационные процессы вовлекалось все больше пользователей, как расширялся круг задач, решаемых с помощью ЭВМ, возникла необходимость в децентрализации процессов обработки данных. Принцип **централизованной** обработки данных (рисунок 1), когда к одной большой ЭВМ подключалось несколько терминалов (рабочих мест, состоящих из дисплея и клавиатуры), не отвечал высоким требованиям к надежности процесса обработки, затруднял развитие систем и не мог обеспечить необходимые временные параметры при диалоговой обработке данных в многопользовательском режиме. Кратковременный выход из строя центральной ЭВМ приводил к роковым последствиям для системы в целом. Поэтому приходилось дублировать функции центральной ЭВМ, значительно увеличивая затраты на создание и эксплуатацию систем обработки данных.

Распределенная обработка данных – обработка данных, выполняемая на независимых, но связанных между собой ЭВМ, представляющих распределенную систему.

Для реализации распределенной обработки данных были созданы **многомашинные ассоциации**, структура которых разрабатывается по одному из следующих направлений:

- многомашинные вычислительные комплексы (МВК);
- компьютерные (вычислительные) сети.

Многомашинный вычислительный комплекс – это группа установленных рядом вычислительных машин, объединенных с помощью специальных средств сопряжения и выполняющих совместно единый информационно-вычислительный процесс.

Многомашинные вычислительные комплексы могут быть:

- **локальными**, при условии установки ЭВМ в одном помещении;
- **дистанционными**, если некоторые ЭВМ комплекса установлены на значительном расстоянии от центральной ЭВМ и для передачи данных используются телефонные каналы связи.

С появлением персональных ЭВМ их также стали объединять в сети.

Компьютерная (вычислительная) сеть – это совокупность компьютеров, соединенных с помощью каналов связи в единую систему, удовлетворяющую требованиям распределенной обработки данных.

Глобальная вычислительная сеть объединяет абонентов, расположенных в различных странах, на различных континентах. Взаимодействие между абонентами такой сети может осуществляться на базе телефонных линий связи, радиосвязи и систем спутниковой связи. Глобальные вычислительные сети позволяют решить проблему объединения информационных ресурсов всего человечества и организации доступа к этим ресурсам. Примером глобальной вычислительной сети служит **всемирная сеть Internet**.

Региональная вычислительная сеть связывает абонентов, расположенных на значительном расстоянии друг от друга. Она может включать абонентов внутри большого города, экономического региона, отдельной страны. Обычно в этом случае расстояние между абонентами вычислительной сети составляет десятки – сотни километров.

Локальная вычислительная сеть объединяет абонентов, расположенных в пределах небольшой территории. В настоящее время не существует четких ограничений на территориальный разброс абонентов локальной вычислительной сети. Обычно такая сеть привязана к конкретному месту. К классу локальных вычислительных сетей относятся сети отдельных предприятий, фирм, банков, офисов и т. д. Протяженность такой сети можно ограничить пределами 2 ... 2,5 км.

2. Локальные вычислительные сети

Назначение всех видов компьютерных ЛВС определяется, в основном, двумя функциями:

обеспечение совместного использования аппаратных и программных ресурсов сети;

обеспечение совместного доступа к ресурсам данных.

Локальные вычислительные сети объединяют относительно небольшое число ПЭВМ (обычно от 10 до 100) в пределах небольшой территории (одно помещение, здание или учреждение). Традиционное название – локальная вычислительная сеть – скорее дань тем временам, когда сети использовались в основном для решения вычислительных задач. Сегодня же в 99% случаев речь идет исключительно об обмене информацией в виде текстов, графических и видео-образов, числовых массивов.

Основными техническими средствами ЛВС являются **рабочие станции** (обычные ПЭВМ), которые могут быть подключены к более мощному компьютеру – **сетевому серверу**.

Сервер – это мощный компьютер, способный одновременно обрабатывать сотни (тысячи) запросов на обслуживание, поступающих от рабочих станций.

Рабочей станцией (или сетевым узлом) называется ПЭВМ, используемая в качестве клиента, то есть потребителя сетевых услуг, предоставляемых сервером.

3. Глобальная вычислительная сеть

Зарождение Интернета принято считать с момента появления первой компьютерной сети, родиной которой в середине 60-х годов двадцатого века стала Америка.

В то время еще не существовало персональных компьютеров, и крупные американские университеты могли себе позволить 1-2 больших компьютера. Компьютерное время было драгоценным ресурсом, и на него заранее записывались. Люди работали ночами, чтобы ни минуты этого времени не пропало даром.

Наконец появилась идея соединить между собой компьютеры разных университетов, чтобы сделать возможным удаленное использование любого свободного в данный момент компьютера. Этот проект получил название ARPANET. К концу 1969 года были соединены компьютеры четырех университетов и появилась первая компьютерная сеть.

Очень скоро обнаружилось, что сеть в основном используется не для вычислений на удаленном компьютере, а для обмена сообщениями между пользователями. В 1972 году, когда ARPANET уже соединял 23 компьютера, была написана первая программа для обмена электронной почтой по сети. Электронную почту оценили по достоинству, что побудило целый ряд государственных организаций и корпораций к созданию собственных компьютерных сетей. Эти сети обладали тем же недостатком, что и ARPANET: они могли соединять только ограниченное число однотипных компьютеров. Кроме того, они были не совместимы друг с другом.

В середине 70-х годов для ARPANET были разработаны новые стандарты передачи данных, которые позволяли объединять между собой сети произвольной архитектуры, тогда же было придумано слово "Интернет". Именно эти стандарты, впоследствии получившие название протокола TCP/IP, заложили основу для роста глобальной компьютерной сети путем объединения уже существующих сетей. Их важным достоинством было то, что сеть считалась в принципе не стопроцентно надежной, и предусматривались средства борьбы с ошибками при передаче данных. В 1983 году сеть ARPANET перешла на новый протокол и разделилась на две независимые сети - военную и образовательную. К этому времени сеть объединяла более тысячи компьютеров, в том числе в Европе и на Гавайских островах. Последние использовали спутниковые каналы связи.

Развитие Интернета получило новый импульс благодаря инициативе Национального научного фонда США (NSF) по созданию глобальной сетевой инфраструктуры для системы высшего образования (1985-88). NSF создал сеть

скоростных магистральных каналов связи и выделял средства на подключение к ней американских университетов, при условии, что университет обеспечивал доступ к сети для всех подготовленных пользователей. Интернет оставался преимущественно университетской сетью до начала 90-х годов, однако NSF сразу взял курс на то, чтобы сделать его в дальнейшем независимым от государственного финансирования. В частности, NSF поощрял университеты к поиску коммерческих клиентов. К 1988 году Интернет уже насчитывал около 56 тысяч соединенных компьютеров.

Настоящий расцвет Интернета начался в 1992 году, когда была изобретена новая служба, получившая странное название «Всемирная паутина» (World Wide Web, или WWW, или просто «веб»). WWW позволял любому пользователю Интернета публиковать свои текстовые и графические материалы в привлекательной форме, связывая их с публикациями других авторов и предоставляя удобную систему навигации. Постепенно Интернет начал выходить за рамки академических институтов и стал превращаться из средства переписки и обмена файлами в гигантское хранилище информации. К 1992 году Интернет насчитывал более миллиона соединенных компьютеров.

В настоящее время Интернет продолжает расти с прежней головокружительной скоростью. По оценке специалистов, количество передаваемой информации (трафик) в Интернете увеличивается на 30% ежемесячно. В 1999 году Интернет объединял около 60 миллионов компьютеров и более 275 миллионов пользователей, и каждый день в нем появлялось полтора миллиона новых веб-документов. Эти оценки довольно приблизительны, потому что в Интернете нет центрального административного органа, который регистрировал бы новых пользователей и новые компьютеры.

В Россию Интернет впервые проник в начале 90-х годов. Ряд университетов и исследовательских институтов приступили в это время к построению своих компьютерных сетей и обзавелись зарубежными каналами связи. Особенно следует отметить Институт Атомной Энергии им. Курчатова. На базе ИАЭ сложились две крупнейшие коммерческие компании, предоставляющие услуги по подключению к Интернету – «Релком» и «Демос», а также Российский Институт Развития Общественных Сетей (РОСНИИРОС). Последний стал в дальнейшем головной организацией, координирующей развитие российской части Интернета.

4. Требования, предъявляемые к современным вычислительным сетям

Производительность

Определяется такими показателями: время реакции системы - время между моментом возникновения запроса и моментом получения ответа. Пропускная способность сети определяется количеством информации, переданной через сеть или ее сегмент в единицу времени. Определяется в битах в секунду.

Надежность

Определяется надежностью работы всех ее компонентов. Для повышения надежности работы аппаратных компонентов обычно используют дублирование, когда при отказе одного из элементов функционирование сети обеспечат другие.

При работе компьютерной сети должна обеспечиваться *сохранность* информации и защита ее от искажений. Как правило, важная информация в сети хранится в нескольких экземплярах. В этом случае необходимо обеспечить согласованность данных (например, идентичность копий при изменении информации).

Одной из функций компьютерной сети является передача информации, во время которой возможны ее потери и искажения. Для оценки надежности исполнения этой функции используются показатели вероятности потери пакета при его передаче, либо вероятности доставки пакета (передача осуществляется порциями, которые называются пакетами).

В современных компьютерных сетях важное значение имеет другая сторона надежности - *безопасность*. Это способность сети обеспечить защиту информации от несанкционированного доступа. Задачи обеспечения безопасности решаются

применением как специального программного обеспечения, так и соответствующих аппаратных средств.

Управляемость

При работе компьютерной сети, которая объединяет отдельные компьютеры в единое целое, необходимы средства не только для наблюдения за работой сети, сбора разнообразной информации о функционировании сети, но и средства управления сетью. В общем случае система управления сетью должна предоставлять возможность воздействовать на работу любого элемента сети. Должна быть обеспечена возможность осуществлять мероприятия по управлению с любого элемента сети. Управлением сетью занимается администратор сети или пользователь, которому поручены эти функции. Обычный пользователь, как правило, не имеет административных прав.

Другими характеристиками управляемости являются возможность определения проблем в работе компьютерной сети или отдельных ее сегментов, выработка управленческих действий для решения выявленных проблем и возможность автоматизации этих процессов при решении похожих проблем в будущем.

Расширяемость и масштабируемость

Любая компьютерная сеть является развивающимся объектом, и не только в плане модернизации ее элементов, но и в плане ее физического расширения, добавления новых элементов сети (пользователей, компьютеров, служб). Существование таких возможностей, трудоемкость их осуществления входят в понятие расширяемости. Другой похожей характеристикой является масштабируемость сети, которая определяет возможность расширения сети без существенного снижения ее производительности. Обычно одноранговые сети обладают хорошей расширяемостью, но плохой масштабируемостью. В таких сетях легко добавить новый компьютер, используя дополнительный кабель и сетевой адаптер, но существуют ограничения на количество подключаемых компьютеров в связи с существенным падением производительности сети. В многосегментных сетях используются специальные коммуникационные устройства, которые позволяют подключать к сети значительное количество дополнительных компьютеров без снижения общей производительности сети.

Прозрачность

Прозрачность компьютерной сети является ее характеристикой с точки зрения пользователя. Эта важная характеристика должна оцениваться с разных сторон.

13.1. Лекция № 18 (2 часа)

Тема: Защита информации

13.1.1. Вопросы лекции:

1. Угрозы безопасности информации и их классификация
2. Принципы создания базовой системы защиты информации
3. Методы и средства защиты информации

13.1.2. Краткое содержание вопросов

1. Угрозы безопасности информации и их классификация

Наряду с интенсивным развитием вычислительных средств и систем передачи информации все более актуальной становится проблема обеспечения ее безопасности. Меры безопасности направлены на предотвращение несанкционированного получения информации, физического уничтожения или модификации защищаемой информации.

Под **угрозой безопасности информации** понимается *действие или событие, которое может привести к разрушению, искажению или несанкционированному использованию информационных ресурсов, включая хранимую, передаваемую и обрабатываемую информацию, а также программные и аппаратные средства.*

Угрозы принято делить на *случайные*, или *непреднамеренные*, и *умышленные*. Источником первых могут быть ошибки в программном обеспечении, выходы из строя аппаратных средств, неправильные действия пользователей или администрации и т.д. Умышленные угрозы, в отличие от случайных, преследуют цель нанесения ущерба пользователям АИТ и, в свою очередь, подразделяются на активные и пассивные.

Пассивные угрозы, как правило, направлены на несанкционированное использование информационных ресурсов, не оказывая при этом влияния на ее функционирование. Пассивной угрозой является, например, попытка получения информации, циркулирующей в каналах, посредством их прослушивания.

Активные угрозы имеют целью нарушение нормального процесса функционирования посредством целенаправленного воздействия на аппаратные, программные и информационные ресурсы. К активным угрозам относятся, например, разрушение или радиоэлектронное подавление линий связи, вывод из строя ПЭВМ или ее операционной системы, искажение сведений в базах данных или в системной информации в компьютерных технологиях и т.д. Источниками активных угроз могут быть непосредственные действия злоумышленников, программные вирусы и т.п.

К основным угрозам безопасности информации относят:

1) *Раскрытие конфиденциальной информации* – несанкционированный доступ к базам данных, прослушивание каналов и т.п. В любом случае получение информации, являющейся достоянием некоторого лица (группы лиц) другими лицами, наносит ее владельцам существенный ущерб.

2) *Компрометация информации* реализуется посредством внесения несанкционированных изменений в базы данных, в результате чего ее потребитель вынужден либо отказаться от нее, либо предпринимать дополнительные усилия для выявления изменений и восстановления истинных сведений. В случае использования скомпрометированной информации потребитель подвергается опасности принятия неверных решений со всеми вытекающими отсюда последствиями.

3) *Несанкционированное использование информационных ресурсов*, с одной стороны является средством раскрытия или компрометации информации, а с другой – имеет самостоятельное значение, поскольку, даже не касаясь пользовательской или системной информации, может нанести определенный ущерб абонентам и администрации. Этот ущерб может варьироваться в весьма широких пределах – от сокращения поступления финансовых средств до полного выхода АИТ из строя.

4) *Ошибочное использование информационных ресурсов* может привести к разрушению, раскрытию или компрометации указанных ресурсов. Данная угроза чаще всего является следствием ошибок, имеющих в программном обеспечении АИТ.

5) *Несанкционированный обмен информацией* может привести к получению одним из них сведений, доступ к которым ему запрещен, что по своим последствиям равносильно раскрытию содержания банковской информации.

6) *Отказ от информации* состоит в непризнании получателем или отправителем этой информации фактов ее получения или отправки. В условиях банковской деятельности это, в частности, позволяет одной из сторон расторгать заключенные финансовые соглашения «техническим» путем, формально не отказываясь от них и нанося тем самым второй стороне значительный ущерб.

7) *Отказ в обслуживании* представляет собой весьма существенную и распространенную угрозу, источником которой является сама АИТ. Подобный отказ особенно опасен в ситуациях, когда задержка с предоставлением ресурсов абоненту может привести к тяжелым для него последствиям. Так, отсутствие у пользователя данных, необходимых для принятия решения, в течении периода времени, когда это решение еще возможно эффективно реализовать, может стать причиной его нерациональных или даже антимонопольных действий.

Наиболее распространенными путями несанкционированного доступа к информации, сформулированными на основе анализа зарубежной печати, являются:

- перехват электронных излучений;
- принудительное электромагнитное облучение (подсветка) линий связи;
- применение подслушивающих устройств;
- дистанционное фотографирование;
- перехват акустических излучений и восстановление текста принтера;
- хищение носителей информации и документальных отходов;
- чтение остаточной информации в памяти системы после выполнения санкционированных запросов;
- копирование носителей информации с преодолением мер защиты;
- маскировка под зарегистрированного пользователя;
- мистификация (маскировка под запросы системы);
- использование программных ловушек;
- использование недостатков языков программирования и операционных систем;
- включение в библиотеки программ специальных блоков типа «Троянский конь»;
- незаконное подключение к аппаратуре и линиям связи;
- злоумышленный вывод из строя механизмов защиты;
- внедрение и использование компьютерных вирусов.

2. Принципы создания базовой системы защиты информации

При разработке АИТ возникает проблема по решению вопросов безопасности информации, составляющей коммерческую тайну, а также безопасности самих компьютерных информационных систем.

Современные АИТ обладают следующими основными признаками:

- наличием информации различной степени конфиденциальности;
- необходимостью криптографической защиты информации различной степени конфиденциальности при передаче данных;
- иерархичностью полномочий субъектов доступа и программ к АРМ, файл-серверам, каналам связи и информации системы;
- организацией обработки информации в диалоговом режиме, в режиме разделения времени между пользователями и в режиме реального времени;
- обязательным управлением потоками информации как в локальных сетях, так и при передаче по каналам связи на далекие расстояния;
- необходимостью регистрации и учета попыток несанкционированного доступа, событий в системе и документов, выводимых на печать;
- обязательным обеспечением целостности программного обеспечения и информации в АИТ;
- наличием средств восстановления системы защиты информации;
- обязательным учетом магнитных носителей;
- наличием физической охраны средств вычислительной техники и магнитных носителей.

Организационные мероприятия и процедуры, используемые для решения проблемы безопасности информации, решаются на всех этапах проектирования и в процессе эксплуатации АИТ.

Существенное значение при проектировании придается предпроектному обследованию объекта. На этой стадии выполняются следующие действия:

- устанавливается наличие секретной (конфиденциальной) информации в разрабатываемой АИТ и оценивается уровень конфиденциальности;

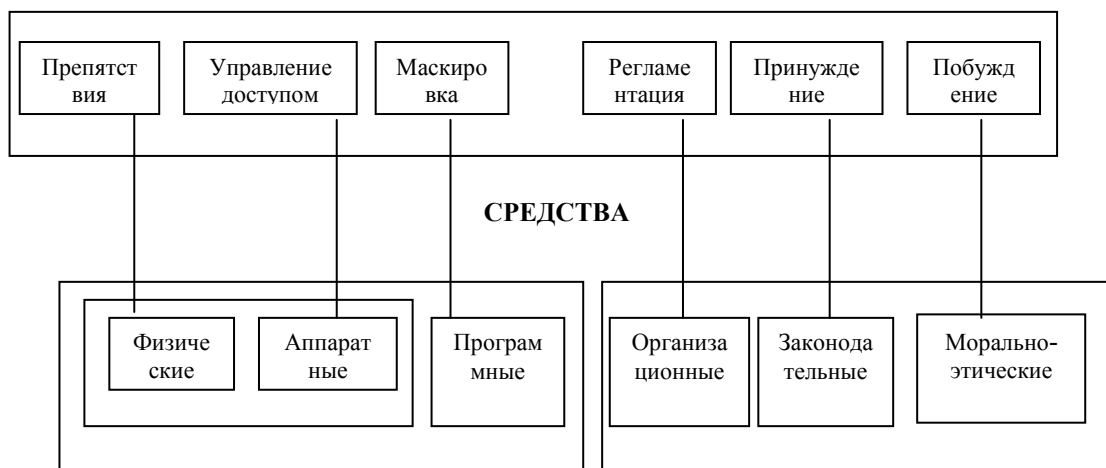
- определяются режимы обработки информации, состав комплекса технических средств, общесистемные программные средства и т.д.;
- анализируется возможность использования имеющихся на рынке сертифицированных средств защиты информации;
- определяется степень участия персонала, функциональных служб, специалистов и вспомогательных работников объекта автоматизации в обработке информации, характер взаимодействия между собой и со службой безопасности;
- определяются мероприятия по обеспечению режима секретности на стадии обработки.

Создание базовой системы защиты информации в АИТ основывается на следующих принципах:

- *Комплексный подход* к построению системы защиты при ведущей роли организационных мероприятий, означающий оптимальное сочетание программных аппаратных средств и организационных мер защиты и подтвержденный практикой создания отечественных и зарубежных систем защиты.
- *Разделение и минимизация полномочий* по доступу к обрабатываемой информации и процедурам обработки, т.е. предоставление пользователям минимума строго определенных полномочий, достаточных для успешного выполнения ими своих служебных обязанностей, с точки зрения автоматизированной обработки доступной им конфиденциальной информации.
- *Полнота контроля и регистрации попыток* несанкционированного доступа, т.е. необходимость точного установления идентичности каждого пользователя и протоколирования его действий для проведения возможного расследования, а также невозможность совершения любой операции обработки информации в АИТ без ее предварительной регистрации.
- *Обеспечение надежности системы защиты*, т.е. невозможность снижения уровня надежности при возникновении в системе сбоев, отказов, преднамеренных действий нарушителя или непреднамеренных ошибок пользователей и обслуживающего персонала.
- *Обеспечение контроля за функционированием системы защиты*, т.е. создание средств и методов контроля работоспособности механизмов защиты.
- *«Прозрачность» системы защиты* информации для общего и прикладного программного обеспечения и пользователей АИТ.
- *Экономическая целесообразность* использования системы защиты, т.е. стоимость разработки и эксплуатации систем защиты информации должна быть меньше стоимости возможного ущерба, наносимого объекту в случае разработки и эксплуатации АИТ без системы защиты информации.

3. Методы и средства защиты информации

Методы и средства обеспечения безопасности информации (на примере банковской системы)



Препятствие – метод физического преграждения пути злоумышленнику к защищаемой информации (к аппаратуре, носителям информации и т.д.).

Управление доступом – метод защиты информации регулированием использования всех ресурсов компьютерной информационной системы банковской деятельности (элементов баз данных, программных и технических средств). Управление доступом включает следующие функции защиты:

- идентификацию пользователей, персонала и ресурсов системы (присвоение каждому объекту персонального идентификатора);
- опознание объекта или субъекта по предъявленному им идентификатору;
- проверку полномочий (проверка соответствия дня недели, времени суток, запрашиваемых ресурсов и процедур установленному регламенту);
- разрешение и создание условий работы в пределах установленного регламента;
- регистрацию (протоколирование) обращений к защищаемым ресурсам;
- реагирование (сигнализация, отключение, задержка работ, отказ в запросе) при попытках несанкционированных действий.

Маскировка – метод защиты информации путем ее криптографического закрытия. Этот метод защиты широко применяется за рубежом как при обработке, так и при хранении информации, в том числе на дискетах. При передаче информации по каналам связи большой протяженности этот метод является единственно надежным.

Регламентация – метод защиты информации, создающий такие условия автоматизированной обработки, хранения и передачи защищаемой информации, при которых возможности несанкционированного доступа к ней сводились бы к минимуму.

Принуждение – такой метод защиты, при котором пользователи и персонал системы вынуждены соблюдать правила обработки, передачи и использования защищаемой информации под угрозой материальной, административной или уголовной ответственности.

Побуждение – такой метод защиты, который побуждает пользователя и персонал системы не разрушать установленные порядки за счет сложившихся моральных и этических норм (как регламентированных, так и неписаных).

Средства, с помощью которых реализуются рассмотренные методы:

Технические средства реализуются в виде электрических, электромеханических и электронных устройств. Вся совокупность технических средств делится на аппаратные и физические. Под аппаратными техническими средствами принято понимать устройства, встраиваемые непосредственно в вычислительную технику или устройства, которые сопрягаются с подобной аппаратурой по стандартному интерфейсу.

Физические средства реализуются в виде автономных устройств и систем. Например, замки на дверях, где размещена аппаратура, решетки на окнах, электронно-механическое оборудование охранной сигнализации.

Программные средства представляют из себя программное обеспечение, специально предназначенное для выполнения функций защиты информации.

Организационные средства защиты представляют собой организационно-технические и организационно-правовые мероприятия, осуществляемые в процессе создания и эксплуатации вычислительной техники и аппаратуры телекоммуникаций для обеспечения защиты информации.

Морально-этические средства защиты реализуются в виде всевозможных норм, которые сложились традиционно или складываются по мере распространения вычислительной техники и средств связи в обществе.

Законодательные средства защиты определяются законодательными актами страны, которыми регламентируются правила пользования, обработки и передачи

информации ограниченного доступа и устанавливаются меры ответственности за нарушение этих правил.

Все перечисленные средства защиты разделены на *формальные* (выполняющие защитные функции строго по заранее предусмотренной процедуре без непосредственного участия человека) и *неформальные* (определяющиеся целенаправленной деятельностью человека либо регламентирующие эту деятельность).

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

2.1 Лабораторная работа № 1 (2 часа).

Тема: «Понятия информации.»

2.1.1 Цель работы: Сформировать навыки и умения перевода чисел из любой системы счисления в десятичную

2.1.2 Задачи работы:

1. Перевод чисел из одной позиционной системы счисления в другую.
2. Арифметические операции с числами в позиционных системах счисления

2.1.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПК, программа калькулятор
2. Калькулятор

2.1.4 Описание (ход) работы:

1. Перевести данное число из десятичной системы счисления в двоичную:

а) 464(10); б) 380,1875(10); в) 115,94(10) (получить пять знаков после запятой в двоичном представлении).

Решение.

464 0	380 0	1875	115 1	94
232 0	190 0	0 375	57 1	1 88
116 0	95 1	0 75	28 0	1 76
58 0	47 1	1 5	14 0	1 52
а) 29 1	б) 23 1	1 0	в) 7 1	1 04
14 0	11 1		3 1	0 08
7 1	5 1		1 1	0 16
3 1	2 0			
1 1	1 1			

а) $464(10) = 111010000(2)$; б) $380,1875(10) = 101111100,0011(2)$; в) $115,94(10) \approx 1110011,11110(2)$ (в настоящем случае было получено шесть знаков после запятой, после чего результат был округлен).

Если необходимо перевести число из двоичной системы счисления в систему счисления, основанием которой является степень двойки, достаточно объединить цифры двоичного числа в группы по столько цифр, каков показатель степени, и использовать приведенный ниже алгоритм. Например, если перевод осуществляется в восьмеричную систему, то группы будут содержать три цифры ($8 = 2^3$). Итак, в целой части будем производить группировку справа налево, в дробной — слева направо. Если в последней группе недостает цифр, дописываем нули: в целой части — слева, в дробной — справа. Затем каждая группа заменяется соответствующей цифрой новой системы. Соответствия приведены в таблицах. Р

00	01	10	11	
4	0	1	2	3

Р	2	000	001	010	011	100	101	110	111
8	0	1	2	3	4	5	6	7	

P	2	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	
1011	1100	1101	1110	1111									
16	0	1	2	3	4	5	6	7	8	9	A	B	C
	D	E	F										

Переведем из двоичной системы в восьмеричную число $1111010101,11(2)$.
 $001\ 111\ 010\ 101,110(2) = 1725,6(8)$.

Переведем из двоичной системы в шестнадцатеричную число $1111010101,11(2)$.
 $0011\ 1101\ 0101,1100(2) = 3D5,C(16)$.

Соответствие между шестнадцатеричными цифрами и десятичными числами													16	0	1	2	3
	4	5	6	7	8	9	A	B	C	D	E	F					
10	0	1	2	3	4	5	6	7	8	9	10	11					
	12	13	14	15													

При переводе чисел из системы счисления с основанием Р в десятичную систему счисления необходимо пронумеровать разряды целой части справа налево, начиная с нулевого, и в дробной части, начиная с разряда сразу после запятой слева направо (начальный номер -1). Затем вычислить сумму произведений соответствующих значений разрядов на основание системы счисления в степени, равной номеру разряда. Это и есть представление исходного числа в десятичной системе счисления.

2. Перевести данное число в десятичную систему счисления.

Некоторые неотрицательные степени числа 2 (в десятичной системе счисления)											Показатель	0	1
2	3	4	5	6	7	8	9	10	11	12	13		
14	15	16											
Степень1	2	4	8	16	32	64	128	256	512	1024	2048		
4096	8192	16384	32768	65536									

Некоторые отрицательные степени числа 2 (в десятичной системе счисления)											Показатель	-1	-2	-
3	-4	-5	-6	-7										
Степень0,5	0,25	0,125	0,0625	0,03125	0,015625	0,0078125								

а) 1000001(2).

$$1000001(2) = 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 64 + 1 = 65(10).$$

Замечание. Очевидно, что если в каком-либо разряде стоит нуль, то соответствующее слагаемое можно опускать.

б) 1000011111,0101(2).

$$1000011111,0101(2) = 1 \times 2^9 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-2} + 1 \times 2^{-4} = 512 + 16 + 8 + 4 + 2 + 1 + 0,25 + 0,0625 = 543,3125(10).$$

Некоторые неотрицательные степени числа 8 (в десятичной системе счисления)											Показатель	0	1
2	3	4											
Степень1	8	64	512	4096									

Некоторые отрицательные степени числа 8 (в десятичной системе счисления)											Показатель	-1	-2
Степень0,125	0,015625												

в) 1216,04(8).

$$1216,04(8) = 1 \times 8^3 + 2 \times 8^2 + 1 \times 8^1 + 6 \times 8^0 + 4 \times 8^{-2} = 512 + 128 + 8 + 6 + 0,0625 = 654,0625(10).$$

Некоторые неотрицательные степени числа 16 (в десятичной системе счисления)											Показатель	0	1
2	3	4											
Степень1	16	256	4096	65536									

2.2 Лабораторная работа № 2 (2 часа).

Тема: «Операционная система MS DOS. Оболочка Norton Commander. Основные команды и принципы работы.»

2.1.1 Цель работы: Познакомиться с видами , назначением, составом и этапами загрузки ОС.

2.1.2 Задачи работы:

1. Различать ОС.
2. Проследить этапы загрузки ОС.

2.1.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. ПК
2. ОС MS DOS
3. Оболочка Norton Commander

2.1.4 Описание (ход) работы:

Среди всех системных программ, с которыми пользователю приходится работать, особое место занимает ОС.

ОС управляет работой компьютера, запускает программы, обеспечивает защиту данных, выполняет различные сервисные функции по запросам пользователей и программ.

Операционная система (ОС) – это комплекс системных программ, обеспечивающий совместное функционирование всех устройств компьютера и поддерживающий работу всех его программ.

Каждая программа пользуется услугами ОС, а потому может работать только под управлением той ОС, которая обеспечивает для нее эту услугу. Таким образом, выбор ОС очень важен, так как он определяет с какими программами вы сможете работать. Так же от выбора ОС зависит производительность ваших работ, степень защиты данных, а также возможности прикладных программ.

Человек в форме команд дает указание ОС, она в свою очередь переводит их на язык, понятной машине и передает ей на выполнение. Компьютер обрабатывает полученную информацию, и результаты передает ОС. ОС переводит их на язык понятный человеку и выводит их на экран.



ОС – это дисковая ОС фирмы Microsoft. Дисковая, потому что загружается в работу с диска (винчестера). Была разработана в 1981 году. Недостатками является то, что эта ОС является однозадачной и однопользовательской. Версия MS DOS 6.22 выпущенная в 1994 году работает как самостоятельная ОС. Версия 7.0 1995 года выпуска работает только в системе Windows., т.е. ее правильнее рассматривать, как подсистему Windows.

Диалог пользователя с DOS осуществляется в форме команд, которые пользователь вводит с клавиатуры в ответ на приглашение DOS. Ввод каждой команды заканчивается нажатием клавиши Enter..Когда ОС готова к приему команд, она выводит приглашение C:\>.

Приглашение, как правило, содержит информацию о текущем дисковом и текущем каталоге.

Например:

A:>\ - текущим является корневой каталог диска A:.

C:>\ - текущим является корневой каталог диска C:.

C:\WINDOW> - текущим является каталог WINDOWS диска C:.

D:\GAME\DOOM> - текущим является подкаталог DOOM каталога GAME диска D:.

Логический диск – это физический диск, реальный диск или часть физического диска, которому присвоено имя.

Способ хранения файлов на дисках компьютера называется файловой системой. Иерархическая структура, в виде которой ОС отображает файлы, и папки диска называют файловой структурой.

Файл – это поименованная область на диске или другом носителе информации. В файлах может храниться различная информация: тексты, рисунки, чертежи, таблицы. Имя файла имеет обозначение, которое состоит из двух частей: собственно имени и расширения. Имя файла всегда уникально, оно образуется не более из 8 символов латинского алфавита. Тип файла (расширение) служит для характеристик, хранящейся в файле информации и образуется не более чем из 3 символов. Например:

com – командный системный файл,

exe – исполняемый файл,

txt – текстовый файл,

hlp – файл справки,

doc – расширение текстового процессора MS Word,

avi – файл содержит видеoinформацию,

html – файл содержит Web-страницу

Расширение от основного имени отделяется точкой, например, text.txt.

Имя файла – последовательность символов, позволяющая пользователю ориентироваться в файловой системе и идентифицировать файлы.

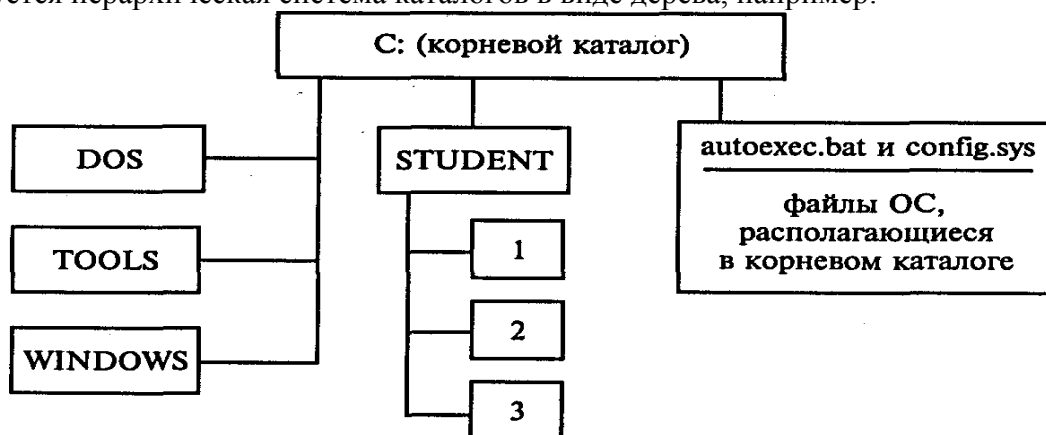
Тип (расширение) файла – последовательность символов, позволяющая компьютеру сопоставить программное обеспечение содержимому файла.

Каталог – это специальное место на диске, в котором регистрируются имена файлов, их месторасположение на диске, а также дополнительные сведения о файлах и их атрибутах. Каталоги имеют свои собственные имена и могут храниться в других каталогах вместе с обычными файлами. Имена каталогов строятся по тем же правилам, что и имена файлов. Например: Student, группа. Расширение для каталогов задавать не принято. Каталог, с которым в настоящее время работает пользователь, называется *текущим*. Каталог, с которым нет связи в данный момент времени, называется *пассивным*.

Чтобы получить доступ к файлу данных, необходимо указать к нему путь. **Путь к файлу** – это последовательность из имен дисков и каталогов, разделенных символом «\».

Полное имя файла – имя логического диска+путь к файлу+имя файла.

Кроме файлов, в каталогах могут содержаться подкаталоги. В результате на диске образуется иерархическая система каталогов в виде дерева, например:



Самым первым является корневой каталог. В данном примере это каталог диска C:

Доступ к файлу организуется с помощью задания пути (цепочка соподчиненных каталогов, которую необходимо пройти по иерархической структуре к каталогу, где зарегистрирован искомый файл). Взаимодействие пользователя с операционной системой осуществляется с помощью командной строки, индицируемой на экране дисплея.

Групповые спецификации

При операциях копирования, перемещения, удаления файлов и т.п. для облегчения работы файлы можно объединять в группы. Для объединения файлов пользуют следующие значки:

* — заменяет любое число символов в имени файла или его расширении;

? — заменяет один произвольный символ.

Обозна	Пояснения к команде
*.doc	Группа файлов, имеющих расширение dos :
n*g.*	Группа всех файлов, имеющих первую букву в имени файла n и последнюю букву g. Расширение может быть любым .
m?h.*	Группа всех файлов, имя файла которых состоит из трех букв, первая из которых m, последняя h. Расширение может быть любым

Список внутренних команд

Внутренние команды MS DOS встроены в командный процессор command.com и работают под его управлением. Команды вводятся с клавиатуры, их ввод завершается нажатием клавиши <ВВОД> (<ENTER>). Набирать команды DOS можно как строчными буквами, так и прописными. В формате записи команд в квадратных скобках помечены необязательные параметры.

1. Команда смены текущего логического диска.

Для смены текущего логического диска надо выбрать имя логического диска, который должен стать текущим, и затем двоеточие, например:

A:

C:.

2. Изменение текущего каталога (переход в другой каталог)

CD <логический диск:> путь

C:\>CD WINDOWS - переход из текущего каталога в каталог WINDOWS. Левая косая черта в начале пути (перед WINDOWS) заставляет идти ОС через корневой каталог

CD.. - переход на уровень вверх

CD\ - переход в корневой каталог текущего диска

3. Просмотр оглавления каталога.

DIR <логический диск:> <путь> <параметры>

DIR - просмотр содержимого в текущем каталоге

A:\ DIR GRUPPA - просмотр содержимого, находящегося на диске A: в каталоге GRUPPA

DIR K*.* - просмотр списка файлов текущего каталога, начинающихся на букву K

DIR *.txt - просмотр списка всех файлов с расширением txt

DIR A?.* - просмотр списка файлов с именами из двух знаков, первый из которых буква A, и произвольными расширениями

DIR /P - просмотр списка файлов «порциями» (постранично)

DIR /W - выводит информацию в сокращенном виде — только имена файлов и директорий (в столбцы).

4. Создание каталога

MD <имя каталога>

C:\ MD LETO - создание каталога под именем LETO в текущем каталоге на текущем диске

C:\ MD STUDENT\NATALI - создание подкаталога с именем NATALI в каталоге STUDENT диска C:

5. Удаление каталога

RD <имя каталога>- если он пустой

DELTREE <имя каталога> - со всеми файлами

RD MY - удаление каталога MY

RD GRAFIK\SPIRAL - удаление каталога SPIRAL. Каталог SPIRAL является подкаталогом каталога GRAFIK.

6. Переименование каталога.

MOVE <старое имя каталога> <новое имя каталога>

MOVE GRUPPA KLAS - переименование каталога GRUPPA в KLAS.

7. Создание текстовых файлов.

Сору con <имя файла>

Сору con pismo.txt - создание текстового файла с именем pismo.txt. Ввести необходимый текст. Для сохранения файла и выхода из режима редактирования текста нажать F6 или Ctrl+Z, затем Enter. Появится сообщение о записи файла на диск в текущей директории и на диске появится файл с указанным именем.

8. Переименование файла.

REN <имя файла 1> <имя файла 2>

REN pismo.doc letter.doc - переименование файла pismo.doc в текущем каталоге в файл letter.doc

9. Вывод содержимого файла на экран.

TYPE <имя файла>

TYPE letter.doc –просмотр файла letter.doc из текущего каталога.

TYPEA:\TEXT\pismo.txt - просмотр содержимого файла pismo.txt, находящегося на диске A: в директории TEXT.

10. Удаление файлов.

DEL <имя файла>

A:\ DEL \NC*.txt - удаление всех файлов с расширением txt из каталога NC диска A:

DEL pismo.txt - удаляет файл pismo.txt

DEL *.psx - удаление всех файлов с расширением psx из текущего каталога

11. Копирование файлов (каталогов).

COPY <имя файла1> <имя файла2>

COPY <имя файла1> <имя каталога2>

COPY <имя каталога1> <имя каталога2>

COPY abc.txt A: - копирование файла abc.txt на диск A:.

C:\ COPY STUDENT\pismo.doc A:\TEKST - копирование файла с именем pismo.doc из каталога STUDENT диска C: в директорию TEKST, находящуюся на диске A:

COPY *.doc A:\ - копирование всех файлов из текущего каталога в корневой каталог диска A:

12. Объединение файлов

COPY <имя файла1> +<имя файла2> <имя файла3>

COPY celeron.txt+athol.txt pentium.txt – объединение файлов celeron.txt и athol.txt в pentium.txt

13. Просмотр дерева каталогов.

C:\TREE – дерево каталогов на диске C:.

TREE LETO дерево каталога LETO.

Команды общесистемного назначения

CLS – очистка экрана

DATE – вывод информации о дате и установка даты в компьютере

TIME – вывод информации о времени и установка времени в компьютер

VER – вывод информации о версии ОС

Внешние команды

Помимо команд, распознаваемых и выполняемых командным процессором в операционной системе имеется большое число утилит — команд, реализованных в виде отдельных программ.

FORMAT [диск:] /f:V /q /s V- задание объема диска. Максимальный объем форматирования для дискет с двумя отверстиями — 1,44 Мбайт, для дискет с одним отверстием — 720 кбайт

FORMAT A: /q - быстрое форматирование дискеты, объем по умолчанию 1,44Мбайт

SYS.com - переносит скрытые системные файлы io.sys, msdos.sys и command.com на требуемый диск

SYS A: - перенос системных файлов из текущей директории (где есть системные файлы) на диск A:

MEM [/c] [/p] - получение информации о распределении памяти компьютера

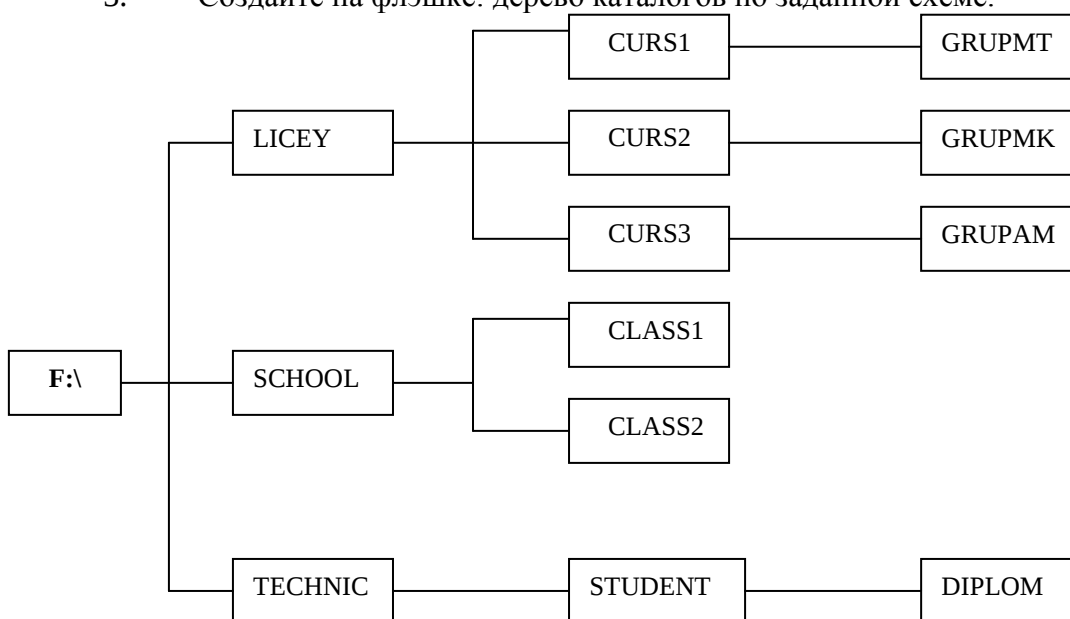
MEM /c /p - вывод постранично на экран списка программ, расположенных в оперативной памяти компьютера, указывая их местоположение и размер (значение параметра в десятичном коде)

MEM - без параметров команды выводит лишь резюме с информацией об имеющемся в распоряжении объеме памяти

PRINT TEXT.txt - вывод на принтер текстового файла с именем TEXT.txt

Выполните следующие задания:

1. Включите компьютер.
2. Запустите MS DOS (Пуск→Программы→Стандартные→Командная строка).
3. Создайте на флэшке: дерево каталогов по заданной схеме.



4. Просмотрите оглавление корневого каталога диска F:.
5. Просмотрите оглавление подкаталога LICEY.
6. Удалите подкаталог GRUPAM.
7. Просмотрите оглавление подкаталога SCHOOL.
8. Удалите подкаталог CLASS1.
9. Создайте текстовые файлы zadania1.txt и zadania2.txt в подкаталоге CLASS2.
10. Просмотрите оглавление подкаталога CLASS2.
11. Переименуйте подкаталог DIPLOM в каталог ПРОЕКТ.
12. Создайте подкаталог ZACHET в каталоге STUDENT.
13. Выведите на экран названия файлов и каталогов, начинающихся на букву L.
14. Переименуйте текстовый файл zadania1.txt в prakt.txt.
15. Просмотрите содержимое файла prakt.txt.

16. Просмотрите содержимое каталога CLASS2.
17. Удалите файл prakt.txt в подкаталоге CLASS2.
18. Скопируйте файл zadania2.txt в подкаталог GRUPMT.
19. Просмотрите содержимое подкаталога GRUPMT.
20. Удалите подкаталог CLASS2.
21. Выведите Дерево каталогов диска F:.
22. Начертите полученную схему Дерева каталогов в тетради для практических работ.
23. Продемонстрируйте результаты работы преподавателю.
24. Завершите работу MS DOS.

2.3 Лабораторная работа №3,4 (4 часа).

Тема: «Операционная система WINDOWS.»

2.3.1.Цель работы: Изложить основные принципы работы операционной системы Windows.

2.3.2 Задачи работы:

1. Изучить базовые технологии работы с основными распространенными операционными системами корпорации Microsoft для персональных IBM-совместимых компьютеров.
2. Владеть основными технологическими приемами работы в операционной системе Windows 98 (XP).

2.3.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.
2. Сервисные программы, ОС Windows.

2.3.4 Описание (ход) работы:

Windows – это название семейства операционных систем для IBM-совместимых персональных компьютеров. Это семейство разработано фирмой Microsoft и предназначено для замены морально устаревшей системы MS DOS. В настоящее время это семейство включает в себя Windows 95, Windows 98 , Windows 2000, Windows XP, Windows NT

Общая характеристика

Windows 98 (2000) является высокопроизводительной, универсальной, надежной многозадачной и многопоточковой интегрированной 32-разрядной операционной системой нового поколения с расширенными сетевыми возможностями, работающей в защищенном режиме (integrated 32-bit protected-mode operating system) обеспечивающей графический интерфейс с пользователем.

Рабочий стол

После загрузки Windows большую часть экрана занимает «Рабочий стол» (Desk Top). На рабочем столе размещаются папки (folder) и значки (icon) основных программ для работы с компьютером, также ярлыки наиболее часто используемых программ, документов и принтеров.

Рабочий стол используется:

- для запуска необходимых программ
- для переключения между программами
- для отображения результатов работы программ в своих окнах

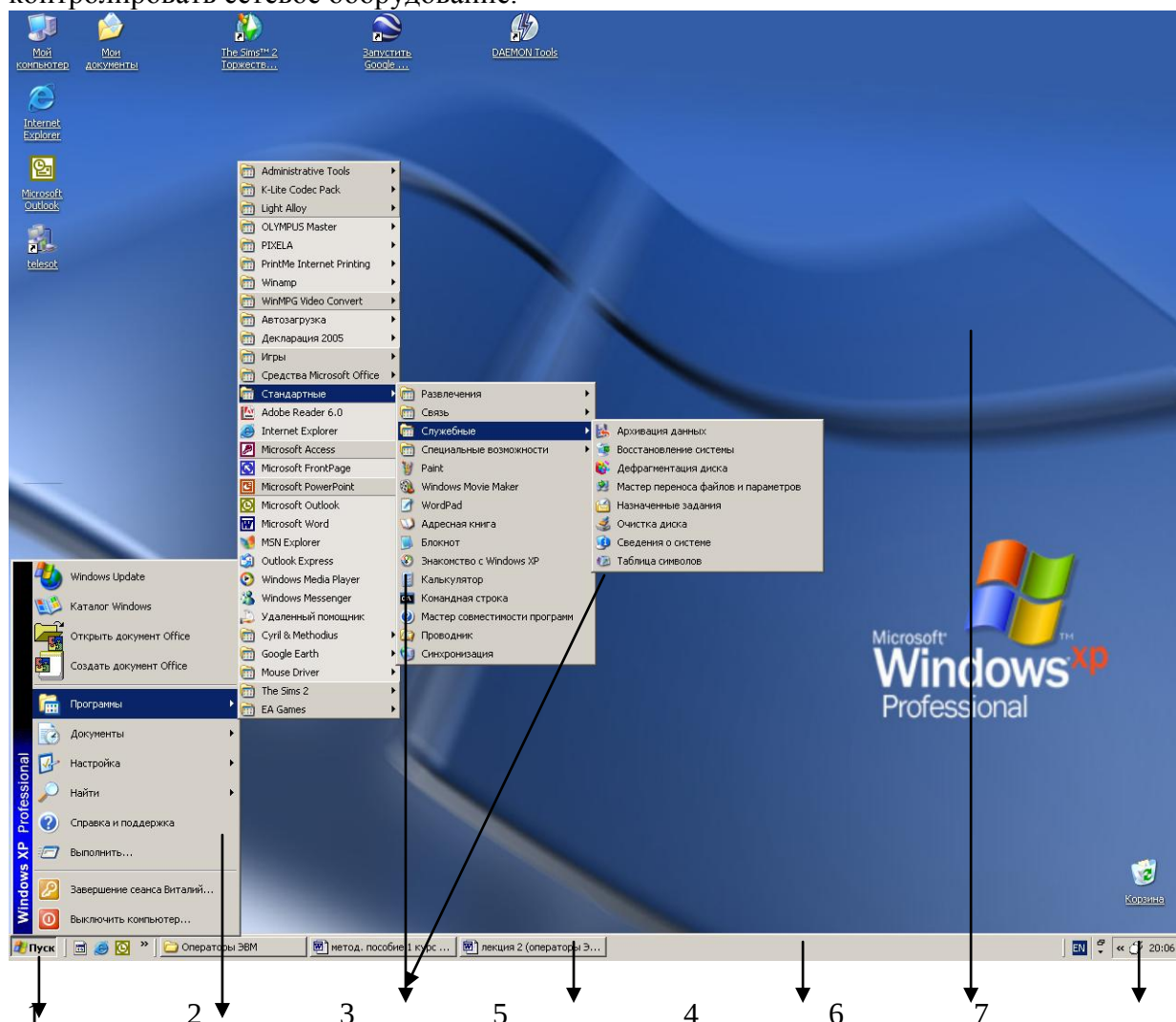
Чаще всего на поверхности стола находятся пиктограммы с подписями:

«**Мой компьютер**» (My Computer) – средства управления компьютером и работы с файлами, хранящимися на встроенных в ПК накопителях:

«**Корзина**» (Recycler) – «мусорная корзина» для сбора ненужных файлов, предназначены для удаления;

«Входящие» - установка средств телекоммуникаций, электронной почты и факса и работа с ними;

«Сетевое окружение» - средства сетевого окружения ПК, позволяющие работать с компьютерными сетями и пользоваться ресурсами сети, а также устанавливать контролировать сетевое оборудование.



1. Кнопка Пуск. Открывает Главное меню.
2. Главное меню – мощный элемент управления, работая с которым можно сделать абсолютно все, что только можно сделать в системе Windows.
3. Вложенное меню – команды, содержащие внутри пункта меню, подменю.
4. Панель задач – элемент управления, который отражает запущенные приложения или открытые папки.
5. Кнопка открытия окон и приложений. Помогает переключаться между приложениями и переходить из одного окна в другое
6. Рабочая область, на которой могут находиться значки приложений, папок, документов и ярлыков.
7. Панель индикации – содержит мелкие значки некоторых индикаторов (индикатор раскладки клавиатуры, индикатор регулятора громкости, индикатор системных часов).

На рабочем столе присутствуют ярлыки. **Ярлык объекта** – это путь к объекту. Сам объект хранится где-нибудь в другом месте. Ярлыки являются средствами быстрого доступа к объектам расположенным в каком-либо каталоге или в сети, к которой подключен компьютер.

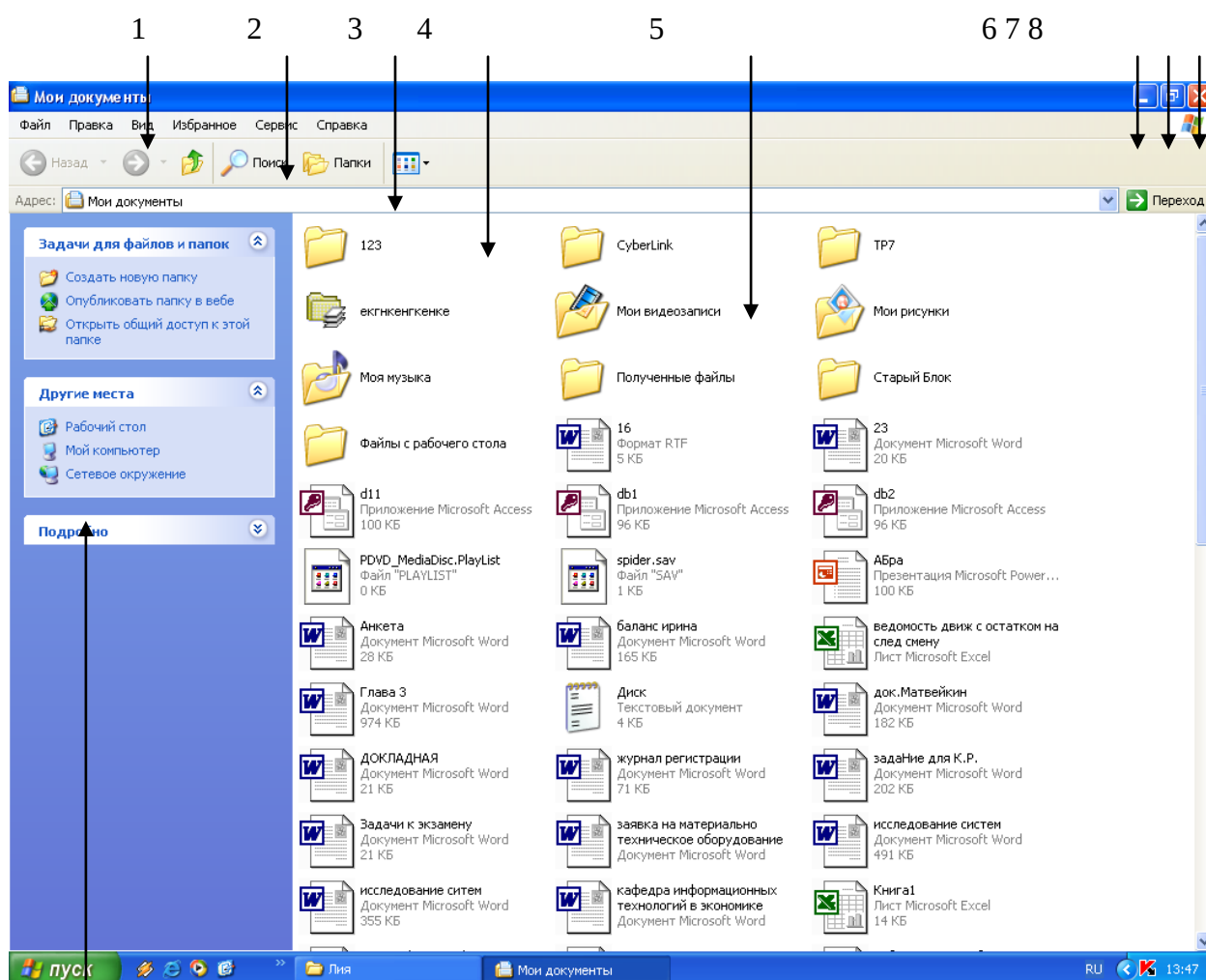
Папки являются средством организации и представления системных ресурсов ПК (каталогов, файлов и т. д.). Папка может содержать другие папки (вложенные папки), программы, также такие объекты, как, например, принтеры и диски. Папка имеет вид закрытого кейса желтого цвета. Пиктограмма – приложение под Windows, связанное со своим индивидуальным значком (таковы значки на «Рабочем столе»).

Виды и свойства окон

В основе системы Windows, также как и других аналогичных систем, лежит понятие *окна*. **Окно** это область экрана, которая связывается с каждой запускаемой программой и используется для управления ее работой. Все окна оформляются в одном стиле и ведут себя одинаково. Одновременно на экране могут находиться несколько окон, которые могут перекрывать друг друга.

Каждое окно имеет рамку и заголовок. Рамка используется для изменения размеров окна с помощью мыши. Заголовок содержит имя запущенной программы и ряд управляющих кнопок. Одна кнопка (maximize) распахивает окно во весь экран, другая (minimize) сворачивает окно до небольшого значка, третья (close) закрывает окно и тем самым завершает программу. Кроме того, заголовок используется для перемещения окна по экрану с помощью мыши и для выделения активного (рабочего) в данный момент окна среди всех открытых окон.

Кроме рамки и заголовка, большинство окон содержат строку меню, панель инструментов, линейки прокрутки, рабочую область, строку состояния.



Элементы окна

1. Строка заголовка – содержит название папки.

2. Строка меню – содержит основные команды, с помощью которых можно управлять объектами, расположенными в папке.
3. Панель инструментов – содержит наиболее часто используемые элементы управления окна. О назначении каждого элемента можно узнать по всплывающей подсказке.
4. Строка адреса – содержит путь к папке.
5. Рабочая область окна – содержит значки объектов, находящихся в папке.
6. Кнопка Свернуть – сворачивает окно на Панель задач.
7. Кнопка Развернуть – разворачивает окно на весь экран.
8. Кнопка Заккрыть – закрывает окно.
9. Информационная панель – позволяет открывать другие папки, выполнять операции с объектами и получать информацию о свойствах объекта.
10. Рамка окна – позволяет изменять размер окна с помощью протягивания мышью.

Изменение размеров окна

1. При помощи кнопок, расположенных в строке заголовка окна:

- кнопка *Свернуть* сворачивает окно в кнопку и помещает ее на панели задач. Чтобы снова развернуть окно, необходимо щелкнуть по данной кнопке;
- кнопка *Развернуть* увеличивает размер окна до максимально возможного;
- кнопка *Восстановить* изменяет размер окна от максимального до среднего и обратно.

2. При помощи мыши:

- подвести указатель мыши к границе окна (вертикальной, горизонтальной, углу), которую необходимо изменить (указатель примет вид двойной стрелки);
- при нажатой левой клавише мыши переместить указатель в нужном направлении.

Способы закрытия окна

1. Комбинация клавиш Alt + F4.
2. Щелчок по кнопке *Заккрыть* строки заголовка окна.
3. Активизировать системное меню и выбрать команду *Заккрыть*. Выполнить команду **Файл/Заккрыть**.

Использование правой кнопки мыши

В Windows для правой клавиши мыши найдено важное оперативное применение – стоит щелкнуть ею, как на месте курсора мышки создается небольшое окно с перечнем ряда операции допустимых для заданного (выделенного в данный момент) объекта. Такое меню называется **контекстным**, т. к. содержание пунктов меню зависит от того, какой объект выделен. Пункты контекстного меню дублируют пункты основного меню.

С помощью контекстного меню очень удобно просмотреть и изменить свойства объекта. Например: настроить цвета экрана в пункте «Свойства», посмотреть количество используемого и свободного пространства на любом диске в пункт «Свойства» контекстного меню. При нажатой правой клавише мыши можно также осуществлять операции копирования или перемещения объектов.

Значок «Мой компьютер»

Одной из наиболее важных пиктограмм на рабочем столе является значок «Мой компьютер» (My Computer). Пиктограмма «Мой компьютер» представляет на рабочем столе папку, как бы содержащую весь компьютер целиком. Этот значок дает доступ к файловой системе конкретного ПК и позволяет запустить любое приложение. Чтобы просмотреть находящиеся в компьютере папки и файлы необходимо дважды щелкнуть значок «Мой компьютер».

Папка «Мой компьютер» содержит значки всех дисковых накопителей ПК, папку «Панель управления» (Control Panel), папку «Принтеры» (Printers). Значки наглядно иллюстрируют назначение того или иного объекта, относящегося к аппаратной части ПК или к его программному обеспечению. Один щелчок левой клавишей мыши приводит к выделению объекта, двойной щелчок запускает закрепленное за значком приложение.

Щелкнув дважды на пиктограмме дисковод «С», можно получить дополнительное окно с содержанием этого диска. Подкаталоги в нем представлены папками, а отдельные файлы значками. Вид пиктограммы файла зависит от того, какой прикладной программой он создавался.

Значок «Корзина»

Значок «Корзина» расположен непосредственно на «Рабочем столе». Фактически это специальный каталог на жестком диске, в котором хранится информация об удаляемых файлах, независимо от того, из какой папки производится удаление. Файлы можно удалять путем перетаскивания пиктограмм файла на «корзину». Корзина позволяет восстановить файлы удаленные по ошибке. После очистки корзины восстановить файлы можно только с помощью специальных программ (утилит), но и это проблематично.

Технология выполнения работы:

Задание 1

1 Разместите значки, находящиеся на Рабочем столе, в произвольном порядке.

2 При помощи контекстного меню Рабочего стола упорядочьте их размещение. Для упорядочения размещения значков на Рабочем столе вызовите контекстное меню. Для вызова контекстного меню щелкните на свободном месте Рабочего стола правой кнопкой мыши.

3 Запустите на выполнение несколько стандартных программ на выбор используя Главное меню: ПУСК/ ПРОГРАММЫ/СТАНДАРТНЫЕ, например, Блокнот, WordPad, Paint, Калькулятор (см. рисунок 1.1).

4 Отобразите окна всех программ в полноэкранном режиме. Для этого необходимо нажать кнопку <Развернуть> .

5 Упорядочите окна всех программ каскадом, слева направо, сверху вниз. Для того чтобы упорядочить размещение открытых окон на Рабочем столе, воспользуйтесь контекстным меню **Панели задач** (т.е. щелкните на свободном месте Панели задач правой кнопкой мыши), опции которого позволяют на Рабочем столе разместить окна, а также определить их свойства: каскадом, слева направо, сверху вниз, свернуть все (показать Рабочий стол).

6 Сверните окно одной из программ, а окно второй восстановите до первоначального размера. Измените с помощью мыши размеры окна. Для переключения программы в различные режимы воспользуйтесь кнопками в правом верхнем углу окна программы. Для регулирования окна вручную подведите к краю окна указатель мыши, который превратится в маркер растяжки. Затем, удерживая нажатой левую кнопку мыши, перетащите маркер в необходимое положение.

7 Закройте открытые программы, кроме документа программы Блокнот. Закройте программу можно одним из способов: щелкнув по кнопке <Закроить>, выполнив команду Выход меню ФАЙЛ, нажав комбинацию клавиш <ALT> и <F4>.

8 Найдите в справочной системе Windows три термина: «мультимедиа», «настройка Главного меню», «прокрутка». Скопируйте по очереди их описание и поместите сводный текст в документ БЛОКНОТ. Для этого:

- запустите программу «Справка», используя Главное меню;
- выберите закладку УКАЗАТЕЛЬ;
- в пустом поле введите ключевое слово и щелкните по кнопке **Вывести** или

Показать;

- выделите содержание справки и скопируйте в буфер обмена при помощи опции контекстного меню Копировать;
- активируйте окно программы Блокнот вставьте скопированный материал в Блокнот;
- сохраните выбранный материал при помощи команды Файл/Сохранить, выберите свою папку и наберите имя файла: **справка. txt**, закрыть все окна.

9 Найти путь нахождения файла **справка.txt**. Для поиска объекта вызовите команду Пуск/Поиск(Найти)/Файлы и папки: в окне **Найти** в закладках **имя и размещение, дата изменения, дополнительно** сделайте соответствующие установки. Сверните окно поиска.

10 Создайте ярлык для файла **справка.txt** и поместите его на Рабочий стол. Запустите программу при помощи созданного ярлыка.

11 Поместите файл справка.txt в Главное меню ПРОГРАММЫ. Запустите файл из Главного меню. Для размещения файла (или программы) в Главном меню Программы необходимо:

- выбрать в Главном меню пункт Настройка;
- выбрать пункт Панель задач и меню «Пуск» и откройте вкладку Настройка меню;
- щелкнуть по кнопке Добавить, а затем - по кнопке Обзор;
- найти нужный файл и дважды щелкните по нему;
- щелкнуть по кнопке Далее и затем дважды щелкните по нужной папке (программы) в структуре меню;
- ввести в поле название новой команды, добавляемой в меню, и щелкните по кнопке Готово.

12 Удалите программу-приложение из меню ПРОГРАММЫ. Для удаления программы из Главного меню или меню Программы:

- выберите в главном меню пункт Настройка;
- выберите пункт Панель задач и откройте вкладку Настройка меню;
- щелкните по кнопке Удалить, а затем найдите в структуре меню удаляемую программу;
- для того чтобы открыть меню, щелкните стоящий рядом с ним знак «+».

Выделите удаляемый пункт меню и щелкните по кнопке Удалить.

Задание 2. Работа в служебной программе «Проводник»

1 Запустите приложение программы «Проводник» одним из способов. Запуск программы Проводник можно выполнить с помощью:

- кнопки Пуск, используя контекстное меню;
- папки Мой компьютер, используя контекстное меню.

2 Выполните с окном программы Проводник (см. рисунок 1.2) стандартные операции: уменьшение размеров окна, перемещение окна в пределах Рабочего стола, сворачивание окна программы Проводник, разворачивание окна программы Проводник.

3 Измените соотношения внутренних окон программы. Для этого необходимо подвести указатель мыши к границе между окнами и левой кнопкой мыши перетащить на нужное расстояние влево или вправо.

4 Изучите состав меню окна Проводник. Ознакомьтесь с командами меню ФАЙЛ и переведите указатель мыши на очередное меню.

5 Ознакомьтесь с названием кнопок Панели инструментов, указывая поочередно указателем мыши на каждую из них. Названия кнопок поочередно появляются под кнопками Панели инструментов.

6 Щелкните по  любой папки, щелкните по  любой папки оцените результат.

7 Отобразите содержимое любой папки, щелкнув по ней. Упорядочите объекты данной папки по имени, типу, размеру или дате.

8 Выберите один из видов отображения папок: плитка (крупные значки), эскизы страниц, список, таблица. Обратите внимание чем отличаются друг от друга отображения.

9 Для получения информации об объекте используйте: команду **Свойства** меню **Файл** или контекстное меню. Получите информацию про файл: справка.txt.

10 Ознакомьтесь с содержанием рабочего логического диска C:, просмотрев все ветви на соответствующей панели программы Проводник, и получите информацию о свойствах логического диска.

11 Откройте личную папку и создайте одним из способов папку **Персональная**.

12 Скопируйте документ **справка.txt** в папку Персональная на Рабочем столе, используя при этом метод перетаскивания.

13 Создайте две копии папки Персональная и поместите одну из них на Рабочий стол, используя при этом метод перетаскивания.

14 Переименуйте копию папки Персональная в личной папке на **Персональная_1** на Рабочем столе в **Персональная_2**.

15 Скопируйте папку Персональная_2 в личную папку, используя при этом метод перетаскивания.

16 Откройте содержимое папки Персональная_2. Создайте в ней несколько разного типа файлов: точечный рисунок, текстовый документ, документ MS Word.

17 Удалите из папки Персональная_1 файл справка.txt.

18 Просмотрите содержимое Корзины. Восстановите только что удаленный файл.

19 Закройте все окна.

Лабораторная работа 5,6, 7,8

Тема: Разработка алгоритмов линейной, ветвящейся и циклической структуры (с параметром, с постусловием, с предусловием).

2.5.1.Цель работы: Научится составлять простейшие алгоритмы

2.5.2 Задачи работы:

1. Изучить методы составления алгоритмов и их свойства.

2.5.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.

2. Доска

2.5.4 Описание (ход) работы:

Задание 1. Построить алгоритмы задач указанных преподавателем.

Вычислить значение переменной Y, если для любых значений исходных переменных A и X.

Изучив поставленную задачу, приходим к выводу, что исходной информацией являются переменные A и B, а результат работы программы – значение переменной Y.

Проанализировав формулу расчета Y приходим к выводу, что здесь линейный вычислительный процесс, так как при любых значениях переменных A и X формула имеет смысл и порядок решения задачи единственный:

1) Ввод в память ЭВМ значений переменных A и X.

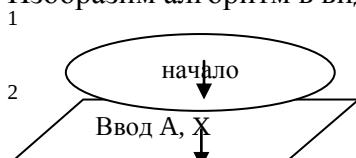
2) Расчет переменной Y.

3) Вывод результата на экран.

Перечень этих пунктов и есть текстовый алгоритм решения задачи.

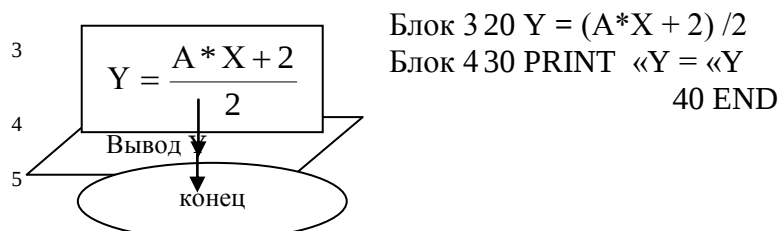
В контрольной задаче можно рассмотреть только один вариант, например: при A=2 и X=3, Y=4.

Изобразим алгоритм в виде блок–схемы:



Описание алгоритма на языке Бейсик

Блок 2 10 INPUT «введите значения переменных A и X», A, X



Анализ работы операторов программы.

Строка 10: по оператору ввода INPUT.

1) На экране дисплея высвечивается приглашение к вводу соответствующее символьной константе оператора:

Введите значения переменных A и X:

2) В оперативной памяти компьютера резервируются две ячейки для хранения значений переменных A и X.

3) Приостанавливается работа программы.

Пользователь должен на клавиатуре набрать две константы, являющиеся значениями переменных через запятую и нажать клавишу ENTER.

4) Переменным A и X присваивается значение, то есть в ячейки памяти, зарезервированные для хранения значений этих переменных, запишутся соответствующие константы. По контрольной задаче, в результате работы оператора Input, получим A=2 и X=3 и программа работает дальше.

Строка 20: по оператору присваивания

1) Резервируется ячейка памяти для хранения значения переменной Y.

2) В арифметическое выражение, вместо имен переменных A и X подставляются их значения.

3) Выполняются арифметические операции с учетом их старшинства и получаем значение арифметического выражения.

4) Значение арифметического выражения присваивается переменной Y, то есть в отведенную для переменной Y ячейку памяти, записывается константа, являющаяся значением арифметического выражения.

Строка 30: По оператору вывода PRINT на экране выводится символьная константа Y= и затем значение переменной Y. Таким образом, по контрольной задаче на экране должно появиться изображение Y=4.

Для закрепления материала, сидя за компьютером, отредактируйте этот текст программы:

1) Описать ввод значений A и X двумя операторами INPUT.

2) Записать в программе оператор PRINT так, чтобы на экране в контрольной задаче появился ответ в следующем виде: Y=4 при A=2 и X=3.

Рассмотренная нами первая задача является одноразовой, т.е. по ней можно ввести только по одному значению переменных A и X и получить только одно значение Y. Если есть необходимость рассчитать несколько значений Y с различными значениями A и X, надо будет каждый ряд запускать программу заново. (Запускать клавишей F2, а не вводить текст заново с клавиатуры).

Рассчитать $Y = (AX + 2) / 2$ для известного количества любых значений A и X, то есть программа должна быть запущена один раз, а получить значений Y столько, сколько нужно. Например, необходимо рассчитать четыре значения переменной Y, со следующими значениями A и X: A=2, -1, 4, 0

X=3, 6, 1, 9

тогда мы должны получить значения

Y=4, -2, 3, 1

№ п/п	A	X	Y
1	2	3	4
2	-1	6	-2
3	4	1	3
4	0	9	1

поскольку мы не изменили формулу
расчета Y, то порядок решения задачи
не изменится:

1) ввод
значений A, X

2) расчет Y

3) вывод Y

Но эти три операции необходимо повторить четыре
раза. Мы знаем, что если какая то группа операций повторяется,
то мы имеем циклический процесс. На первых порах не
пытаться разработать алгоритм циклического процесса
наскоком. Действуйте постепенно, опираясь на теорию, и
начните с того, что вспомните классификацию циклических
процессов с тем, чтобы определиться, какой циклический
процесс организовать в данной задаче:

Так как в задаче № 2 четко сказано, что количество
значений переменных A и X известно, то логично сделать вывод:
необходимо организовать явный циклический процесс по счетчику.

Итак, исходные данные для решения этой задачи: количество значений A, X и сами их значения.

Введем условные обозначения:

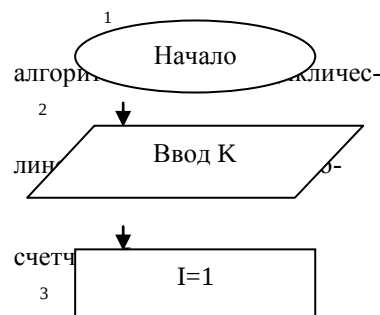
K - количество значений переменных A, X и Y

I – порядковый номер значения переменных

В контрольной задаче K=4 и I=1, 2, 3, 4

В общем случае I=1, 2, ..., K

Изобразим алгоритм решения задачи в виде блок – схемы:

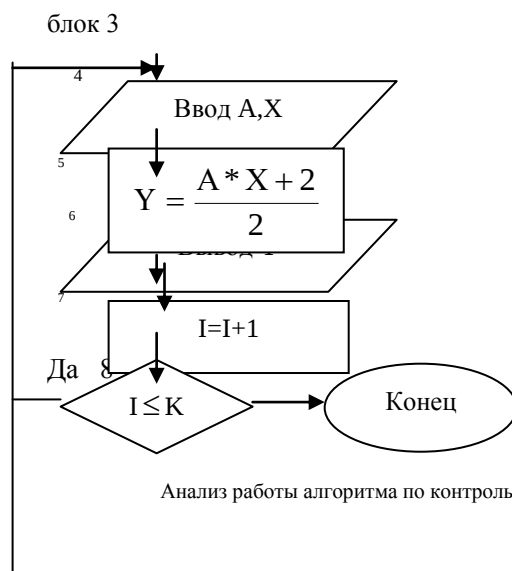


Структура

кий процесс с

ванный по

Вход в цикл –



Анализ работы алгоритма по контрольной задаче:

Номера блоков Результат работы блоков

2 K=4
3 I=1
4 A=2, X=3
5 Y=4
6 Y=4
7 I=1+1=2
8 2 ≤ 4? Да

A=-1, X=6 A=4, X=1
Y=-2 Y=-2
I=2+1=3 I=3+1=4
3 ≤ 4? Да 4 ≤ 4? Да

A=0, X=9
Y=3 Y=3
I=4+1=5
5 ≤ 4? Нет

Выход
из цикла

При анализе работы алгоритма
ваше внимание постоянно должно быть
обращено к контрольной задаче.

На вставочке «нет» из восьмого блока мы оказались
при I=5. Опишем этот алгоритм с помощью операторов языка
Бейсик двумя способами.

1 способ (с помощью операторов переходов):

Бл.2 10 INPUT «Количество значений A, X»;
К
Бл.3 20 I=1
Бл.4 30 INPUT «Введите значения A, X»; A,
X
Бл.5 40 Y=(A*X+2)/2
Бл.6 50 PRINT «Y=»;Y
Бл.7 60 I=I+1

Бл.8 70 IF I ≤ K THEN 30
Бл.9 80 END

Проанализируем работу некоторых операторов программы по контрольной
задаче:

Строка 10: В результате работы оператора INPUT исходная переменная K
получает значение K=4.

Строка 20: В результате работы оператора присваивания переменная I получает
значение равные 1.

Строка 30: В результате операции ввода переменные A и X получают значения.

Строка 60: Происходит переприсваивание значения переменной I на 1 (старое
значение стирается из ячейки памяти).

Строка 70: Работает оператор условного перехода: проверяется условие $I \leq K$?,
если условие выполняется, то работает оператор безусловного перехода на строку с
номером 30, то есть на начало тела цикла, иначе переход на следующую строку, то есть –
выход из цикла.

2.5. Лабораторная работа № ЛР-9,10,11

Тема: Программирование на АЯ высокого уровня

2.5.1.Цель работы: Научится составлять простейшие программы на АЯ высокого

уровня

2.5.2 Задачи работы:

1. Изучить основные операторы АЯ.

2.5.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.

2.5.4 Описание (ход) работы:

Задание 1. Реализовать алгоритмы на АЯ высокого уровня, рассмотренные на ЛР-5,6,7.

2.6 Лабораторная работа 12, 13 (ЛР-12, ЛР-13)

Тема: Массивы. Обработка одномерных массивов на АЯ высокого уровня.

2.6.1.Цель работы: Научится составлять простейшие программы на АЯ высокого уровня

2.6.2 Задачи работы:

1. Изучить основные операторы АЯ.

2.6.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.

2.6.4 Описание (ход) работы:

Задание 1. Реализовать алгоритмы на АЯ высокого уровня, рассмотренные на ЛР-8,9.

2.7 Лабораторная работа №14 (2 часа).

Тема: «Сервисное ПО.»

2.6.1.Цель работы: Изложить основные принципы работы операционной системы Windows.

2.6.2 Задачи работы:

1. Изучить базовые технологии работы с основными распространенными операционными системами корпорации Microsoft для персональных IBM-совместимых компьютеров.

2. Владеть основными технологическими приемами работы в операционной системе Windows 98 (XP).

2.6.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.
2. Сервисные программы, ОС Windows.

2.6.4 Описание (ход) работы:

Задание 1. Отформатируйте 2 дискеты, задав им метки *диск 1* и *диск 2*

1. Выделите значок **Диск 3,5 (A:)** в папке **Мой компьютер**. Если содержимое диска отображается в папке **Мой компьютер**, отформатировать диск будет невозможно. Диск может быть отформатирован только в том случае, если на нем нет открытых файлов.

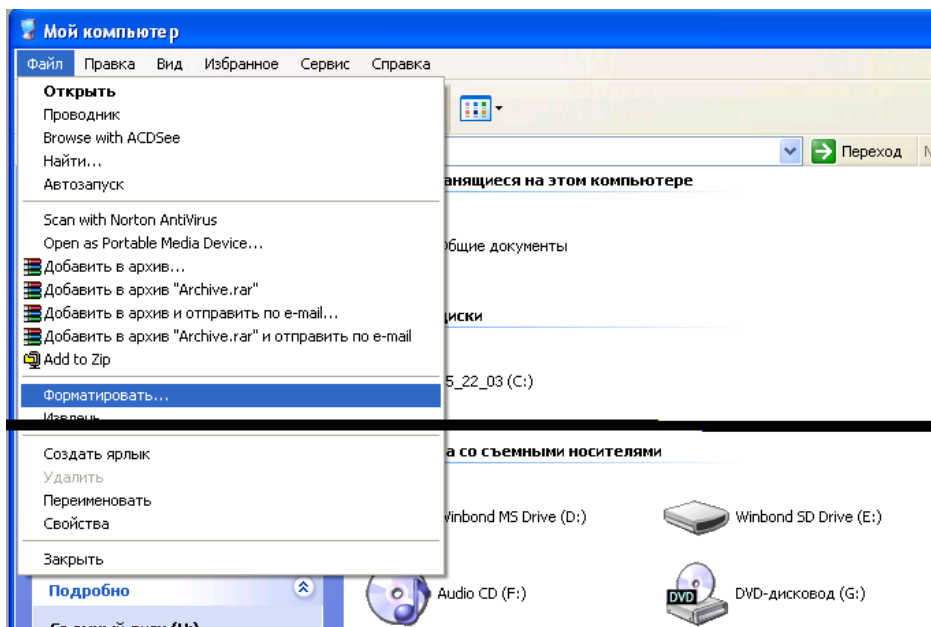


Рисунок 2.1 - Вид меню при выборе пункта Форматировать

2. Выберите команду **Форматировать** в меню **Файл** (см. рисунок 2.1) или в контекстном меню.

3. В открывшемся диалоговом окне Форматирование установите вид форматирования **Полное** и запишите метку дискеты **диск 1** в соответствующем окне. Поставьте флажок в опции **Вывести отчет в результатах**.

4. **Помните!!! Форматирование приводит к полному уничтожению всех данных на диске.**

5. Повторите аналогичные действия со второй дискетой, задав метку дискеты **диск 2**.

2 Скопируйте на первую дискету несколько различных объектов (папок, файлов, ярлыков, приложений). Для копирования данных на дискету можно воспользоваться любым способом: командами меню, кнопками Панели инструментов, методом *drag and drop*. Наиболее быстрым и удобным способом является команда контекстного меню *Отправить, Диск 3,5(A)*.

Задание 2. Проверьте поверхность дискеты диск 1 и информацию, содержащуюся на ней, программой Проверка диска. Получите итоговую информацию о сканировании

1 Необходимо запустить программу проверки диска. Для этого необходимо в папке **Мой компьютер** выделить диск **A:**.

2 В контекстном меню выбрать **Свойства**. В диалоговом окне **Свойства диска** (рисунок 2.2) выбрать вкладку **Сервис** и пункт **Выполнить проверку**.

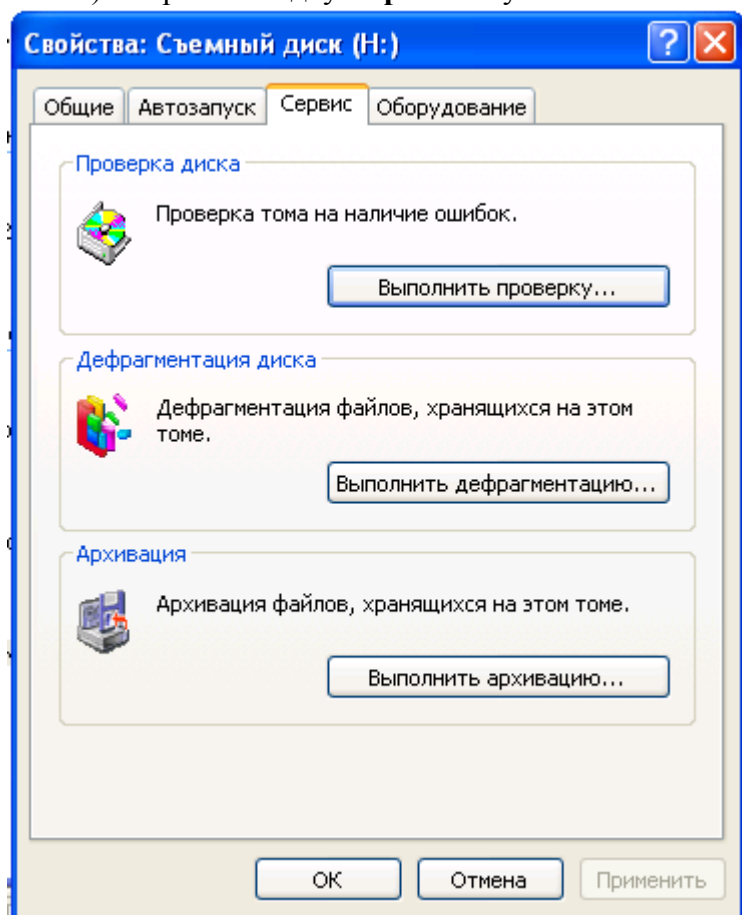


Рисунок 2.2 - Вид окна «Свойства диска».

3 Установите параметры проверки в соответствии с рисунком 2.3 и нажмите кнопку **Запуск**.

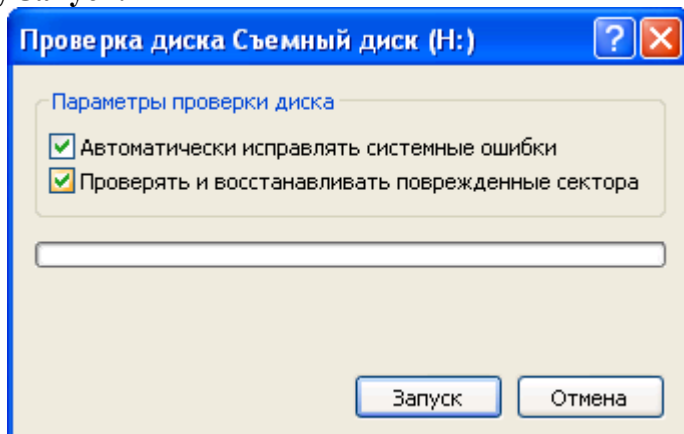


Рисунок 2.3 - Параметры проверки диска.

Задание 3. Выполните дефрагментацию дискеты *диск 1*

1 Необходимо запустить программу дефрагментации диска. Для этого необходимо в папке **Мой компьютер** выделить **диск A:**.

2 В контекстном меню выбрать **Свойства**. В диалоговом окне **Свойства диска** (рисунок 2.2) выбрать вкладку **Сервис** и пункт **Выполнить дефрагментацию**.

3 Установите параметры фрагментации и нажмите **Старт**.

Задание 4. Сделайте копию дискеты *диск 1* на дискете *диск 2*

Создание копии диска позволяет перенести все объекты с *диска 1* на *диск 2*, не используя промежуточные операции, что экономит время и ресурсы.

Для создания копии необходимо выполнить следующие действия:

1 Вставьте в дисковод дискету *диск 1*.

2 Выделите значок Диск 3,5 (A:) в папке Мой компьютер или на правой панели программы Проводник. Выберите в меню **Файл** команду **Копировать диск**.

3 В открывшемся окне **Копирование диска** (рисунок 2.4) щелкните по кнопке **Начать**.

4 После чтения содержимого *диск 1* и копирования его в буфер последует предложение о вставке диска 2 (рисунок 2.5). Вставьте *диск 2* и щелкните по кнопке **ОК**.

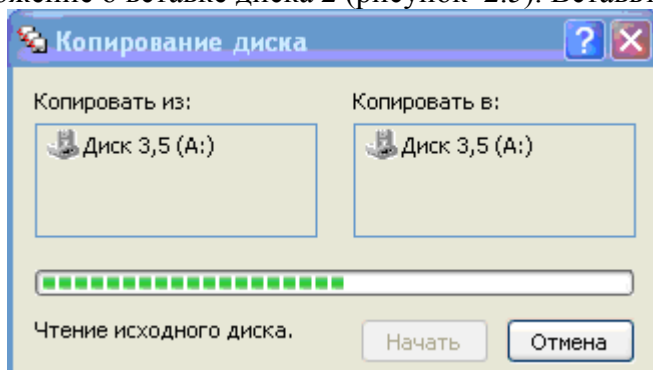


Рисунок 2.4 - Окно «Копирование диска».

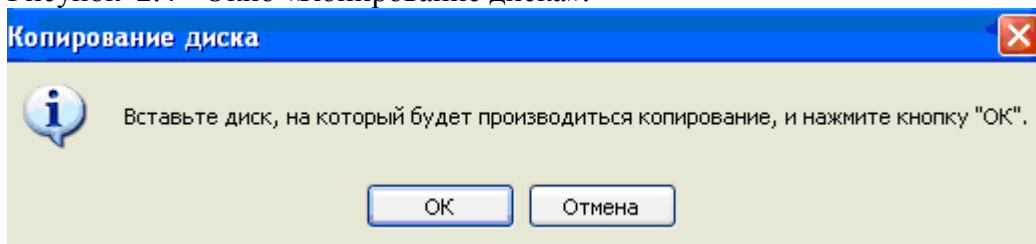


Рисунок 2.5 - Сообщение после копирования содержимого первого диска.

Задание 5. Получите информацию о диске *диск 2*

Для того чтобы получить информацию о диске необходимо:

- выделите необходимый диск и выберите команду **Свойства** в меню Файл или контекстного меню;
- просмотреть все необходимые вкладки, согласно рисунка 2.6.

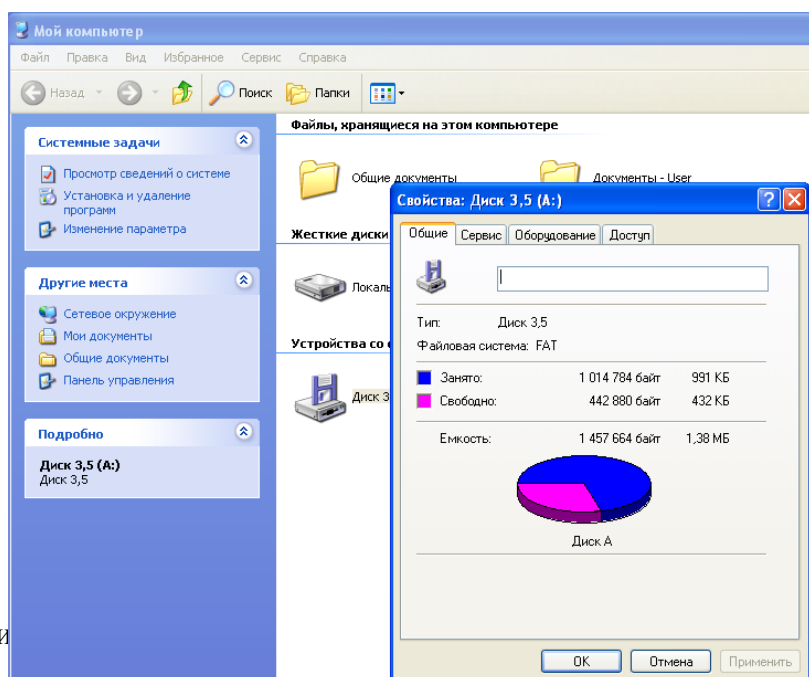


Рисунок 2.6 – Вид

Задание 6. Архивирование файлов

1 Подготовьте объекты архивирования. Для этого необходимо создать в личной папке папку **Архив_1**. В эту папку поместите несколько различных объектов, например, 5 файлов формата документ MS Word, имеющих расширение .doc, и 5 графических файлов с расширением .bmp, .jpg.

2 Запустите программу **WinRar**. В его рабочем окне (см. рисунок 2.7) найдите и откройте папку **Архив_1**.

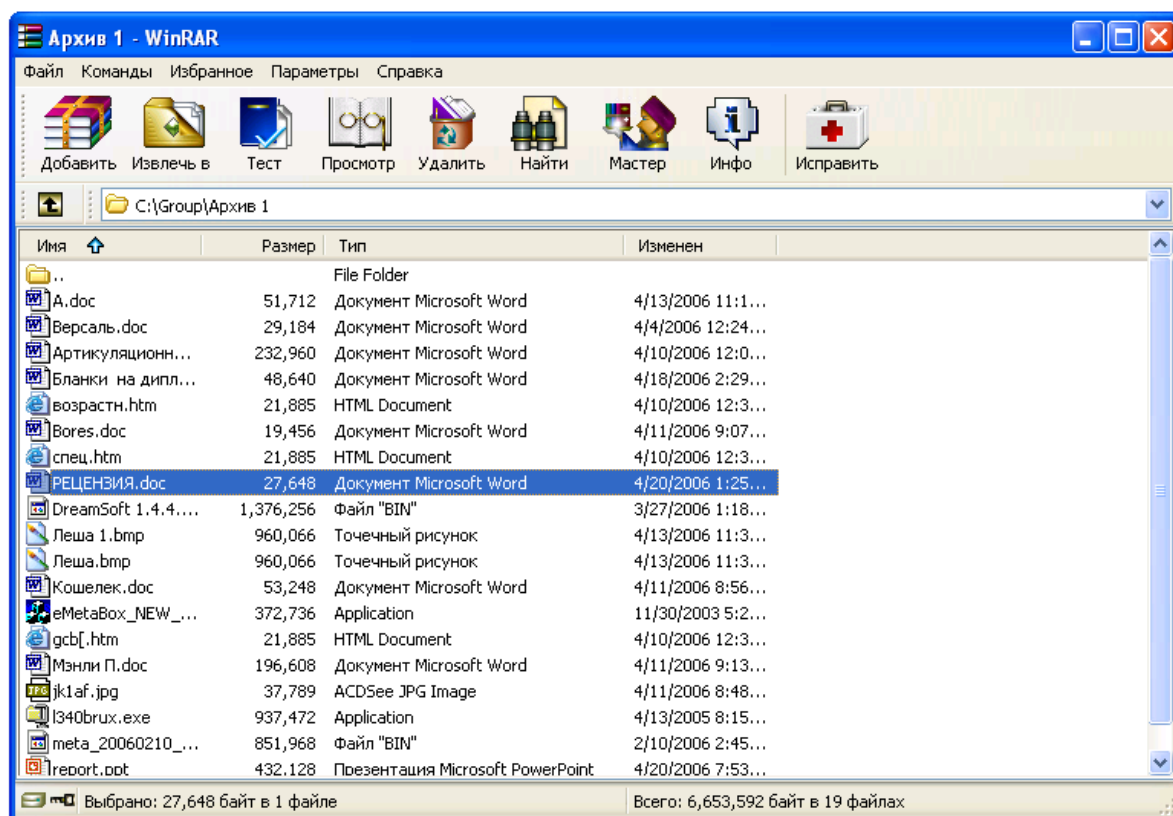


Рисунок 2.7 – Вид рабочего окна программы WinRAR

3 Выделите файлы формата **Microsoft Word**. Для этого можно воспользоваться следующим способом: выбрать команду меню **Файл Выделить группу** (рисунок 2.8).

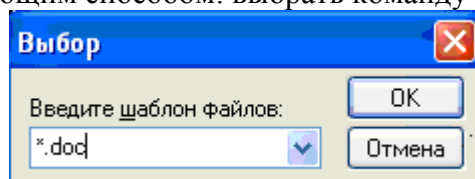


Рисунок 2.8 - Окно выделения группы файлов

4 Заархивировать данные файлы. Для этого можно воспользоваться командой меню **Команды, Добавить файлы в архив** (рисунок 2.9) или кнопкой на Панели инструментов **Добавить** или командой контекстного меню **Добавить файлы в архив**.

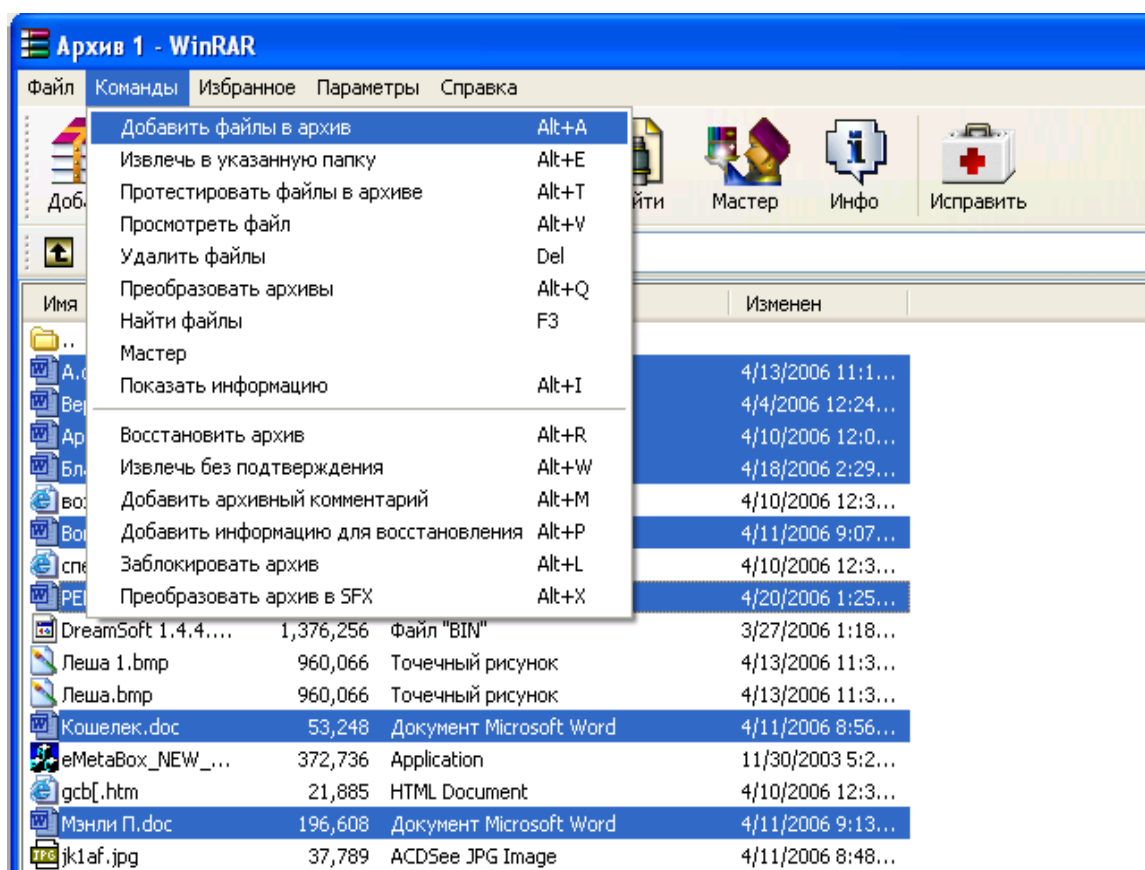


Рисунок 2.9 - Добавление файлов в архив

2.8 Лабораторная работа №15 (2 часов).

Тема: «Прикладное ПО.»

2.8.1.Цель работы: получить начальные навыки работы с текстовым документом. Изучение информационной технологии создания документов MS Word с использованием Шаблонов. Использование списков.

2.8.2 Задачи работы:

1. Создание документов в MS WORD
2. Редактирование документов в MS WORD

2.8.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Персональный компьютер.
2. Текстовый процессор Microsoft Word

2.8.4 Описание (ход) работы:

Запуск MS Word и создание нового документа

Запуск программы Word может осуществляться любым из стандартных способов запуска приложений:

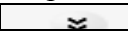
- Щелкните кнопку **Пуск** и выберите в меню пункт **Создать документ Microsoft Office**. В открывшемся окне дважды щелкните по значку **Новый документ**.
- Щелкните кнопку **Пуск**, выберите пункт **Программы, Microsoft Word**.
- Дважды щелкните левой клавишей мыши по значку **Microsoft Word** на Рабочем столе.

Как правило, при запуске программы MS Word можно сразу приступить к вводу текста. Создание нового документа в других случаях (уже в самом текстовом редакторе) осуществляется командами **Файл, Создать** или с помощью кнопки **Создать**  на панели инструментов Стандартная.

Экранный интерфейс редактора Microsoft Word (Окно программы MS Word)

После запуска программы на экране появляется окно с открытым в нем пустым документом, которому по умолчанию присваивается имя **Документ 1**. Окно содержит следующие элементы (Рис.1):

Заголовок окна (первая строка) – слева в строке заголовка находится значок Word (системное меню), рядом имя загруженного файла документа, с которым вы работаете в текущий момент (в данном случае этот файл называется **Документ 1**), затем – имя программы - **Microsoft Word** и справа 3 кнопки управления окном (свернуть, развернуть (свернуть в окно), закрыть документ).

Строка меню – находится под строкой заголовка. Каждый пункт меню раскрывается и имеет соответствующее подменю, если по нему щелкнуть левой клавишей мыши. Пункты строки меню открываются в два приема. На первом этапе открываются сокращенные меню, а затем (при необходимости, щелкнув по кнопке ) - расширенное меню.

Панель инструментов – находится ниже строки меню. Обычно на экране, по умолчанию, отображаются две панели инструментов *Стандартная* и *Форматирование*. Остальные панели при необходимости могут быть открыты командами: **Вид, Панели инструментов**. Внутри панелей инструментов собраны различные элементы управления, которые представляют наиболее часто выполняемые операции при работе с документами. Панели инструментов обеспечивают быстрый и более простой доступ к важнейшим функциям редактора.

Ниже находится Рабочая область – это область экрана, где отображается создаваемый документ и производится его редактирование. Сверху и слева документа находятся горизонтальная и вертикальная линейки.

Линейка – используется при формировании абзацев и таблиц. Для вывода на экран линейки, нужно выполнить команды: **Вид, Линейка**.

Справа и внизу документа – полосы вертикальной и горизонтальной прокрутки. Они применяются в тех случаях, когда весь текст не помещается на экране и требует сдвига (прокрутки) вверх-вниз или влево-вправо, используя при этом *кнопки прокрутки* или *ползунков прокрутки*.

Окно документа Word можно разделить на две части (с помощью команды меню **Окно, Разделить**) и работать одновременно с двумя частями одного и того же документа. При этом каждая часть документа имеет свою собственную *полосу прокрутки*. Полосу разделения можно перемещать с помощью мыши.

Разделить экран на два подокна можно перетаскиванием вниз маленькой кнопки (вешки полосы разделения), которая находится над верхней стрелкой вертикальной полосы прокрутки.

Закрыть второе подокно можно командой меню **Окно, Снять разделение** или перемещением разделителя.

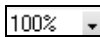
Мигающая вертикальная черта (штрих) - текстовый курсор – указывает, куда в следующий момент времени будет вводиться знак текста, вставляться таблица или рисунок. Передвигать текстовый курсор можно с помощью клавиш управления курсором или с помощью мыши, произведя щелчок в нужном месте документа.

Нижняя строка окна программы называется информационной строкой или строкой состояния. В ней отображается различная информация о самом документе и о текущем состоянии текстового процессора. Первые четыре числа означают номер страницы в разделе, номер раздела, номер страницы в документе и сколько всего страниц.

Первоначальные сведения и правила работы с Word.

1. При работе с документом пользователь задает подходящий режим просмотра и редактирования текста. Режим устанавливается с помощью пункта меню

Вид. В режиме **Обычный** достигается наибольшая скорость ввода и редактирования текста. Режим **Разметка страницы** более точно отображает окончательный внешний вид страниц документа при печати, кроме того, он обязателен при работе с графическими объектами, рамками и многоколоночным текстом. Режим **Структура документа** и **Режим электронного документа** полезны опытным пользователям при работе с большими документами.

2. Используя команды **Вид, Масштаб** можно изменять масштаб видимости документа. Масштабирование текста можно также осуществлять с помощью комбинированного поля *Масштаб*  на панели инструментов *Стандартная*.

3. Начинать работу с текстовым документом необходимо с настройки параметров страницы. Это осуществляется с помощью команды меню **Файл, Параметры страницы**. В диалоговом окне **Параметры страницы** выбрав вкладку **Поля**, можно установить отступы от всех краев листа бумаги. Вкладка **Размер бумаги** позволяет установить размер самого листа бумаги и его ориентацию при печати: **Книжная** или **Альбомная** и т.д.

4. Так же можно пронумеровать страницы. Для этого нужно:

- выполнить команду меню **Вставка, Номера страниц**;
- указать положение и выравнивание номера.

Номера страниц можно установить, используя колонтитулы. Колонтитулы представляют собой специальное поле, размещаемое, как правило, в верхнем и нижнем полях страницы, содержащее различные объекты, повторяющиеся на каждой странице, например, время, имя файла, имя автора и т.п. Устанавливаются колонтитулы через команду меню **Вид, Колонтитулы**.

5. Для автоматического переноса слов в документе необходимо выполнить команды **Сервис, Язык, Расстановка переносов** и в появившемся диалоговом окне включить переключатель *Автоматическая расстановка переносов*.

6. Для проверки правописания, необходимо щелкнуть кнопку **Правописание** на панели инструментов *Стандартная* или в меню **Сервис** выбрать команду **Правописание**.

Основные приемы редактирования текста

Под редактированием понимается изменение уже существующего документа. Начинают редактирование с загрузки (открытия) документа. Для этого используется команда **Файл, Открыть** или кнопка **Открыть**  на панели инструментов *Стандартная*, по выполнению которой на экране появляется стандартное диалоговое окно **Открытие документа**. Если нужный документ находится в другой папке, его надо разыскать и открыть.

Процедура редактирования текста включает следующий комплекс операций:

1. **Ввод текста** – производится обычным способом при помощи клавиатуры. Следует отметить, что клавиша Enter служит для обозначения конца абзаца. Перенос текста на новую строку производится автоматически при достижении текстом правой границы. Если при вводе текста была допущена ошибка, она устраняется удалением символов при помощи клавиш Backspace (удаляет символ слева от курсора) или Delete (удаляет символ справа).

2. **Копирование.** Чтобы скопировать фрагмент текста, надо:

- выделить нужный фрагмент;
- скопировать его в буфер обмена с помощью команды меню **Правка,**

Копировать или использовать кнопку  на панели инструментов *Стандартная*;

- установить курсор в то место текста, куда нужно вставить фрагмент;

– выполнить команду **Правка, Вставить** или использовать кнопку  на панели инструментов *Стандартная*. Вставить фрагмент из буфера обмена можно столько

раз, сколько нужно.

Прежде чем начинать редактировать или форматировать фрагмент текста, необходимо его выделить. Это можно сделать с помощью мыши или используя клавиатуру.

Чтобы выделить	Действие
Слово	Дважды щелкнуть слово.
Строку текста или несколько строк текста	Переместить указатель мыши в зону выделения (Зона выделения находится между левой границей текста и листа. В этой зоне указатель мыши принимает вид стрелки, направленной в правый верхний угол ↗) и щелкнуть левой кнопкой мыши. Для выделения более чем одной строки, необходимо удерживая левую кнопку мыши, осуществлять ее перемещение вверх или вниз.
Абзац	Переместить указатель мыши в зону выделения и сделать двойной щелчок левой кнопкой мыши.
Весь текст	Переместить указатель мыши в зону выделения и сделать тройной щелчок левой кнопкой мыши.
Часть строки	Переместите указатель мыши к первому выделяемому символу строки так, чтобы он принял вид I, удерживая нажатой левую кнопку мыши, протяните мышь вправо (влево) до нужного символа.
Блок текста	Щелкните начало фрагмента и удерживая клавишу Shift нажатой, щелкните конец фрагмента.

Чтобы снять выделение, щелкните по любому месту левой кнопкой мыши.

3. **Удаление текста.** Для удаления фрагмента текста необходимо его выделить, а затем выполнить команду **Правка, Вырезать**.

4. **Отмена** последнего действия осуществляется с помощью команд **Правка, Отменить** или с помощью кнопки  на панели инструментов *Стандартная*. Вернуть отмененное действие можно кнопкой **Вернуть** .

Форматирование документа в MS Word

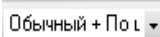
Целью обработки любого текста является придание ему некоторого вида облегчающего восприятие информации и подчиненного общепринятым правилам и стандартам. **Форматирование** – придание документу определенной формы. Форматирование текста осуществляется средствами меню **Формат** или с помощью панели инструментов *Форматирование*. Процедура форматирования текста включает следующие операции:

Форматирование абзацев. При вводе текста новый абзац начинается после очередного нажатия клавиши Enter. Форматирование абзаца подразумевает: выравнивание абзаца по заданным границам, установку межстрочных интервалов и отступов. Для форматирования одного абзаца его не обязательно выделять. Достаточно установить курсор в любое место внутри абзаца. Чтобы применить одинаковое форматирование к нескольким абзацам, их необходимо предварительно выделить.

Выравнивание. Это способ расположения строк между правой и левой границами текста. Существует четыре вида выравнивания абзаца – по центру, влево, вправо и по ширине. Выравнивание абзаца можно выполнить, используя команды **Формат, Абзац**, вкладка **Отступы и интервалы**, в поле **Выравнивание** или кнопки    на панели инструментов *Форматирование*.

Установка межстрочного интервала. Это расстояние между строками в абзаце. Интервалы устанавливаются с помощью команд **Формат, Абзац**, вкладка **Отступы и интервалы**, в комбинированном поле **Межстрочный**.

Форматирование символов – подразумевает изменение типа, стиля, размера, начертания шрифта.

Установка шрифтов. Текст документа может быть оформлен разными шрифтами. Шрифтовое оформление текста производится либо по мере того, как он вводится, либо уже на введенном тексте. Прежде чем выполнять шрифтовое оформление существующего текста, необходимо произвести его выделение. Затем выполнить команды **Формат, Шрифт** или воспользоваться кнопками   на панели инструментов **Форматирование** и выбрать нужные параметры.

Имеется ряд дополнительных возможностей при выборе шрифта: Подчеркивание, Цвет, Эффекты.

Сохранение документа

Для сохранения документа используются команды: **Файл, Сохранить** или **Сохранить как**. Команда **Сохранить** или кнопка  на панели инструментов **Стандартная** сохраняет документ с текущим именем, а команда **Сохранить как** позволяет сохранить документ под новым именем. Для этого в диалогом окне **Сохранение документа** необходимо выбрать *папку*, в которой нужно сохранить документ, ввести *имя файла* с клавиатуры и щелкнуть по кнопке «Сохранить»

Печать документа

Перед тем как печатать документ, его необходимо просмотреть. Для переключения в режим предварительного просмотра необходимо выбрать команды: **Файл, Предварительный просмотр** или использовать на панели инструментов **Стандартная** кнопку . Выход из режима просмотра документа осуществляется щелчком по кнопке «Закрыть».

Печать документа производится командами: **Файл, Печать**. В окне диалога **Печать** выбирают принтер (щелкнув по кнопке **Свойства**, можно изменить свойства печати, например, задать качество печати), указывают номера распечатываемых страниц и количество копий документа. Затем нажимают на кнопку «ОК».

ЗАДАНИЕ 1. Наберите текст, приведенный ниже. В процессе набора текста меняйте начертание, размер шрифта и интервал шрифта.

У В А Ж А Е М Ы Е Г О С П О Д А !

Приглашаем вас на юбилейную презентацию

компьютерной фирмы «МАКУЛШИРС».

Мы работаем на российском рынке много лет. Программные продукты нашей фирмы знают и любят многие пользователи страны. Наши компьютеры работают без рекламаций!

Б у д е м р а д в и д е т ь в а с .

Запомните наш адрес и время нашей презентации

Компьютерная улица, д.5, 18-00.

Справки по телефону 123-45-67

Выполните следующие действия:

1. Выделите ПРИГЛАШЕНИЕ и заключите надпись в рамку. Для этого заключите текст в рамку и произведите цветовую заливку.

Для этого:

- выделить весь текст одним из возможных способов;
- в меню **ФОРМАТ** выбрать **ГРАНИЦЫ И ЗАЛИВКА**;

- во вкладке **ГРАНИЦА** установить тип **РАМКА** и тип линии, **цвет авто**;
- во вкладке **ЗАЛИВКА** установить тип узора (например, 30%) и любой цвет;
- нажмите кнопку ОК.

Замечание: если после рамки предполагается какой-либо текст, то прежде чем выполнять действия по установке рамки необходимо несколько раз нажать <Enter>.

2. Прodelайте ряд самостоятельных упражнений по копированию перестановки и удалению абзацев, предложений в абзацах, слов в предложениях и букв в словах. Для этого в меню **ПРАВКА**, используйте команды: **ВЫРЕЗАТЬ**, **КОПИРОВАТЬ**, **ВСТАВИТЬ**.

3. Проверьте орфографию, используя встроенный словарь: установив курсор в начале текста, выполните команду **СЕРВИС**, **ПРАВОПИСАНИЕ** (при отсутствии ошибок будет выдано сообщение об окончании проверки правописания; при подозрении на ошибку будет выделено соответствующее слово или предложение, которое можно отредактировать, используя кнопки диалогового окна).

4. Выделяя части текста, используйте **ФОРМАТ**, **ШРИФТ**, чтобы поменять цвет слов.

5. В меню **ФОРМАТ**, **ШРИФТ** освоите вкладки **ИНТЕРВАЛ** и **АНИМАЦИЯ**.

6. Освойте функцию поиска и замены. Для этого в меню **ПРАВКА**, **НАЙТИ** и **ЗАМЕНИТЬ**, найти слово Компьютер и заменить на ЭВМ.

7. Используйте в написании документа сноску, используя меню **ВСТАВКА**, **ССЫЛКА**, **СНОСКА**.

8. Примените в тексте буквицу. Для этого в меню **ФОРМАТ**, выбрать вкладку **БУКВИЦА**.

9. Сохраните документ, скопируйте его на свою дискету.

Лабораторная работа 13 (ЛР-13) Создание и редактирование таблиц, формул и графических объектов в MS Word. Создание текстовых документов на основе шаблонов. Создание шаблонов и форм.

СОЗДАНИЕ И РЕДАКТИРОВАНИЕ ТАБЛИЦ В MS WORD

Чтобы создать таблицу в том месте, где находится курсор, достаточно:

- Выбрать в меню **ТАБЛИЦА** опцию **ДОБАВИТЬ ТАБЛИЦУ (ВСТАВИТЬ, ТАБЛИЦА)**. В открывшемся диалоговом окне установить размерность таблицы (число столбцов и строк).

- Быстро создать таблицу можно с помощью кнопки  **Добавить таблицу** на панели инструментов *Таблицы и границы*. Для определения конфигурации новой таблицы нужно закрасить при нажатой левой кнопке мыши требуемое число столбцов и строк таблицы.

- Можно создавать таблицы, рисуя их «карандашом» с помощью мыши при активизированной кнопке  **Нарисовать таблицу**.

Новая таблица состоит из строк и столбцов с пустыми ячейками. Чтобы ввести данные (текст, числа, графика, рисунки, формулы) в ячейку таблицы, щелкните на ячейке (в ней появится текстовый курсор) и наберите нужные данные.

Совет: Если в момент создания таблицы вы еще не знаете, сколько строк вам понадобится, просто создайте таблицу из одной строки. Можно очень легко добавить к концу таблицы новые строки во время ввода текста.

Меню **ТАБЛИЦА** открывает также альтернативные возможности для вставки и изменения таблиц, кроме того, меню позволяет выполнять некоторые дополнительные операции, которые невозможны при использовании интерактивных приемов.

Команды меню «Таблица» и их действия

Команды	Описание
<i>Нарисовать таблицу</i>	<i>Создание новой таблицы</i>
<i>Добавить (Вставить)</i>	<i>Добавление таблицы, строк, столбцов или ячеек</i>
<i>Удалить</i>	<i>Удаление таблицы, выделенных строк, столбцов или ячеек</i>
<i>Выделить</i>	<i>Выделение строки, столбца, ячеек или таблицы целиком</i>
<i>Объединить ячейки</i>	<i>Объединение нескольких выделенных ячеек в одну</i>
<i>Разбить ячейки</i>	<i>Разбиение выделенных ячеек на заданное число строк или столбцов</i>
<i>Разбить таблицу</i>	<i>Разбиение таблицы на две</i>
<i>Автоформат</i>	<i>Автоматическое форматирование таблицы. В списке представлены различные стили оформления таблицы</i>
<i>Автоподбор</i>	<i>Автоматическая настройка ширины и высоты столбцов</i>
<i>Заголовки</i>	<i>Использование выделенной строки в качестве заголовка таблицы, который при печати будет выводиться на каждой странице, если таблица располагается на нескольких листах</i>
<i>Преобразовать</i>	<i>Преобразование таблицы в текст или наоборот</i>
<i>Сортировка</i>	<i>Упорядочивание данных в выделенных строках или списке по алфавиту, величине или дате</i>
<i>Формула</i>	<i>Выполнение математических операций в таблице</i>
<i>Отображать сетку</i>	<i>Команда позволяет включать и отключать отображение сетки из точечных линий</i>
<i>Свойства таблицы</i>	<i>Задание различных параметров таблицы</i>

Вводить данные в таблицу, форматировать и редактировать их по своему усмотрению можно с помощью мыши. Перед тем, как вставлять, удалять или регулировать размеры таблицы, строки, столбца, ячейки необходимо предварительно их выделить с помощью мыши или команд меню **Таблица, Выделить строку** (столбец, ячейку, всю таблицу).

Способы выделения элементов таблицы с помощью мыши

Выделяемый элемент	Способ выделения
Ячейка	Подвести указатель мыши в левый угол ячейки, чтобы он принял вид направленной вправо стрелки ➔, и щелкнуть левой клавишей мыши
Строка	Щелкнуть левой клавишей мыши на полосе выделения – слева от выделяемой строки
Столбец	Установить указатель мыши в верхней части столбца так, чтобы он принял вид направленной вниз черной стрелки ⬇, и щелкнуть левой клавишей мыши
Вся таблица	Нажать левую кнопку мыши в начале таблицы и протащить указатель мыши до конца таблицы

Чтобы поменять **ширину колонок**, надо взяться мышью за правую ее границу или за движок координатной линейки (указатель мыши примет вид ↔) и отодвинуть на нужное расстояние. Чтобы поменять **высоту строки**, надо взяться мышью за горизонтальную линию, отмечающую нижнюю границу строки и подвинуть ее вверх или

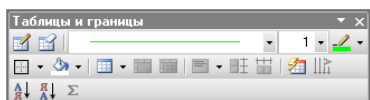
вниз. Чтобы поменять **ширину выделенной ячейки**, надо взяться мышью за правую или левую ее границу и подвинуть в нужную сторону. Она подвинет соседей и увеличится (уменьшится).

Для **выравнивания** текста в ячейке таблицы **по вертикали** следует установить курсор в нужную ячейку, нажать правую кнопку мыши и выбрать в контекстном меню ячейки команду **Выравнивание** и выбрать нужный шаблон.

Для изменения **направления текста** используется команда контекстного меню **Направление текста**.

Для получения **рамки** внутри и вокруг таблицы необходимо выделить таблицу и использовать:

- команды меню **Формат, Границы и заливка**;
- кнопки панели **Форматирование Внешние границы** ;
- кнопки панели инструментов **Таблицы и границы**



. Использование панели инструментов **Таблицы и границы** упрощает работу с таблицами и позволяет реализовывать некоторые дополнительные возможности их форматирования.

ЗАДАНИЕ 1. Создать следующую таблицу (тип шрифта - Times New Roman, 14; межстрочный интервал - одинарный):

Наименование продукции	Посев, площадь, га	Урожайность, ц/га
Ячмень	760	15,4
Просо	300	12,6
Гречиха	210	10,4
Подсолнечник	340	20,6
Пшеница	560	12,5
Рожь	410	16,3

ЗАДАНИЕ №2. Скопируйте таблицу. Преобразуйте скопированную таблицу в ниже следующую, используя всевозможные способы **редактирования**:

Наименование продукции	Посев, площадь, га		Урожайность, ц/га		Недо бор продукции
	200	200	200	200	
	5г.	6г.	5г.	6г.	
Ячмень	760	750	15, 4	15, 5	0,01
Просо	300	250	12, 6	12, 4	0,2
Гречиха	210	240	10, 4	11, 2	0,009
Подсолне чник	340	500	20, 6	19, 0	1,5
Пшеница	560	490	12, 5	12, 3	0,12
Рожь	410	350	16, 3	15, 4	1,01

Добавьте итоговую строку сводных результатов. В итоговой строке рассчитать общую посевную площадь за 2005 и 2006 годы. Для этого установить курсор в ячейку и выполнить команду **ТАБЛИЦА, ФОРМУЛА**. в диалоговом окне «Формула» набрать следующее выражение: **=SUM(RIGHT)**.

Итого	2580	2580			
-------	------	------	--	--	--

Дополнительные задания:

1. Удалите любую строку из таблицы.
2. Добавьте произвольную по содержанию строку в таблицу (после первой строки ниже шапки таблицы).
3. Исклучите 2 столбец (посевная площадь за 2005 год).
4. Отработайте команду ВЫСОТА И ШИРИНА ЯЧЕЙКИ.
5. Измените шрифт и фон шапки таблицы.
6. Добавьте строку в конце таблицы с произвольным содержанием.
7. Измените цветовое оформление таблицы.
8. Сохраните документ на свою дискету.
9. Для закрепления материала создайте таблицу в Приложении (задание №2).

ТЕМА №3.

ЗАПИСЬ МАТЕМАТИЧЕСКИХ ФОРМУЛ

Цель: научиться создавать и редактировать формулы и математические выражения с использованием программного блока Microsoft Equation 3.0 и библиотеки символов.

Для создания и редактирования формул используется программный блок Microsoft Equation 3.0. выполните следующие действия

Краткая справка. На верхней панели (математических символов) расположены кнопки для вставки в формулу более 150 математических символов, большая часть которых недоступна в стандартном шрифте Symbol. Для вставки символов в формулу нажмите кнопку в верхнем ряду панели инструментов, а затем выберите определенный символ из палитры, появляющейся над кнопкой.

На нижней панели (шаблонов) расположены кнопки, предназначенные для вставки шаблонов или структур, включающих символы типа дробей, радикалов, сумм, интегралов, произведений, матриц и различных скобок или соответствующих пар символов типа круглых и квадратных скобок. Во многих шаблонах содержатся специальные места, в которые можно вводить текст и вставлять символы. В редакторе формул содержится около 120 шаблонов, сгруппированных в палитры. Шаблоны можно вкладывать один в другой для построения многоступенчатых формул.

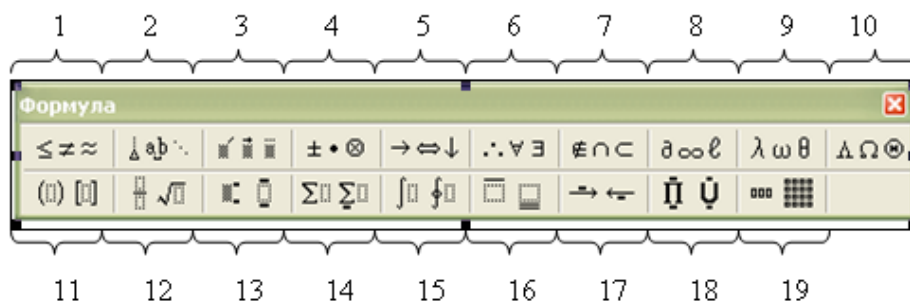


Рис. 3.1. Панель Equation Editor

Назначение нижних и верхних кнопок панели «Редактора формул»

(согласно нумерации кнопок панели на рис. 5.1)

- 1 – вставка символов отношений;
- 2 – вставка пробелов и многоточий;
- 3 – надсимвольные элементы, позволяющие добавлять к математическим переменным примы, крышки, черту или точку;
- 4 – вставка операторов;
- 5 – вставка стрелок;

- 6 – вставка логических символов;
 7 – вставка символов теории множеств;
 8 – вставка разных символов (символы дифференциального исчисления, символы градуса, угла, перпендикуляра и др.);
 9 – вставка строчных букв греческого алфавита;
 10 – вставка прописных букв греческого алфавита;
 11 – вставка шаблонов разделителей:

$$\left(\frac{g}{r}\right) = \frac{g!}{r!(g-r)!};$$

- 12 – вставка шаблонов дробей и радикалов:

$$\sqrt{\frac{1}{z}} + \sqrt{\frac{1}{z} + \sqrt{z}};$$

- 13 – создание верхних и нижних индексов:

$$\lim_{h \rightarrow 0} h(x);$$

- 14 – создание сумм:

$$\sum_{\substack{1 \leq x \leq m \\ 1 \leq y \leq n}} a_{xy} \quad \sum_{i=1}^m \sum_{s=1}^n a_{is} \quad \sum_i \sum_j \sum_k a_{ij} b_{jk} c_{ki};$$

- 15 – вставка интегралов:

$$\Phi(\alpha, \beta) = \int_0^\alpha \int_0^\beta e^{-(n^2+y^2)} dx dy;$$

- 16 – создание математических выражений с чертой сверху и снизу:

$$\overline{\lim_{n \rightarrow 0} h(n)} \quad \underline{\lim_{n \rightarrow 0} h(n)};$$

- 17 – создание стрелок с текстом:

$$f_n(x) \xrightarrow{\text{равномерно}} f : X \xrightarrow{\text{на}} Y;$$

- 18 – вставка произведений и шаблонов теории множеств;

- 19 – вставка шаблонов матриц. Шаблоны этой палитры позволяют создавать векторные столбцы, определители, матрицы и другие макеты типа таблиц:

Функция	Производная
x^n	nx^{n-1}
$\lg x$	x^{-1}
e^x	e^x

ЗАДАНИЕ №1. Создайте следующий фрагмент формулы $\sum_{I=1}^K Z$. Для этого:

- поместите курсор в точку вставки формулы и выполните **Вставка, Объект, вкладка Создание**;
- в поле *Тип объекта* выберите **Microsoft Equation 3.0, ОК** (открыли редактор формул);
- в появившейся *Панели инструментов* выберите **Шаблоны сумм**;
- выберите **Шаблон суммы с верхним и нижним пределами** (крайний правый шаблон в верхнем ряду);
- введите нужный символ, число или выражение в каждый из слотов, предварительно помещая туда курсор;

- Завершите запись формулы (щелчок за пределами рамки или **Esc**).

ЗАДАНИЕ №2. Создайте формулу остатка ряда Тейлора, для этого:

- поместите курсор в точку вставки формулы;
- откройте редактор формул;
- в слоте ввода формулы введите **R**;
- выберите **Шаблоны верхних и нижних индексов**;
- выберите соответствующий шаблон (второй в верхней строке) и введите **n**;
- поместите курсор в конец введенной формулы и введите **=**;
- выберите **Шаблоны дробей и радикалов** и выберите соответствующий шаблон (слева сверху);
- в слот числителя введите $f(\xi)$ (греческую букву выберите на панели инструментов);
- установите курсор справа от **f**;
- выберите **Шаблоны верхних и нижних индексов** и соответствующий шаблон (сверху слева);
- введите **(n+1)**;
- в слот знаменателя введите **(n+1)!**;
- установите курсор в конец введенной формулы и введите оставшуюся часть формулы аналогичным способом;
- завершите запись формулы.

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x-a)^{(n+1)}$$

Для изменения формата формулы выполните следующие действия:

- выделите формулу (одинарный щелчок);
- выберите **Формат, Объект**, вкладка **Обтекание, Вокруг рамки, ОК**;
- увеличьте размер формулы и переместите формулу так, чтобы они не перекрывали друг друга.

ЗАДАНИЕ №3. Создайте формулы с фигурными скобками по образцу:

- откройте редактор формул;
- в слоте ввода введите **y=**;
- выберите **Шаблоны скобок** и нажмите соответствующую скобку;
- поместите курсор в слот и выберите **Шаблоны матриц**, выберите шаблон второй в верхней строке, в верхний и нижний слоты введите соответствующие выражения;
- завершите запись формул.

$$y = \begin{cases} a + x, \text{ при } x > 0 \\ a - x, \text{ при } x \leq 0 \end{cases}$$

ЗАДАНИЕ №4. Создайте матричную формулу по образцу:

- откройте редактор формул;
- введите левую часть формулы (**греческие буквы прописные**)=;
- выберите **Шаблоны матриц** и нажмите шаблон справа внизу;
- в открывшемся диалоговом окне задайте:
- число строк – **3**, число столбцов – **3**;
- выравнивание столбцов – **по центру**, выравнивание строк – **по опорной**

$$\Delta = \begin{vmatrix} a_1 & b_1 & c_1 \\ a_1 & b_1 & c_1 \\ a_1 & b_1 & c_1 \end{vmatrix}$$

линии;

- щелчком слева и справа от образца установите **тип линии** матрицы, подтвердите **ОК**.
- в первый слот первой строки введите **a**;
- выберите **Шаблоны верхних и нижних индексов**, нажмите шаблон, устанавливающий нижний индекс справа и введите **1**;
- аналогичным образом заполните остальные слоты, завершите запись формулы.

Краткая справка Для составления достаточно простых математических выражений можно использовать в меню **Вставка** вкладку **Символ**. Далее в окне диалога **Символ** выбрать шрифт и нужный символ. Выбранный символ вставляется в текст на место курсора за счет нажатия кнопки **Вставить**.

ЗАДАНИЕ.№5. Составьте математическое выражение, используя символичный набор формул.

$$\lim f(x)=A \Leftrightarrow \forall \varepsilon > 0 \exists \delta > 0$$

Для этого:

- установите курсор в то место текста, куда нужно вставить символ;
- выберите в меню команду **Вставка, Символ**, вкладка **Символы**;
- в поле *Шрифт* установите тип шрифта **Symbol**, выберите нужный символ и нажмите кнопку «Вставить».

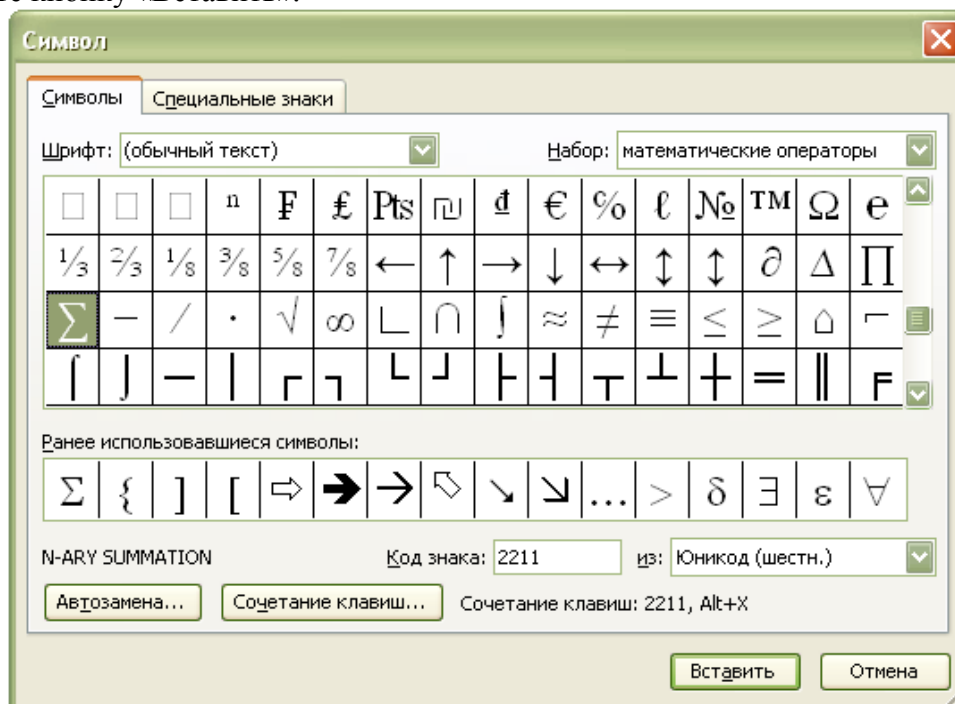


Рис. 3.2. Вставка символа суммы Σ

Сохраните документ под именем **Формулы**.

ЗАДАНИЕ № 6. Для закрепления материала создайте формулу, предложенную в Приложении (задание №3)

ТЕМА 4:

СОЗДАНИЕ БЛОК-СХЕМ В MICROSOFT WORD

В папке «Мои документы», в своей собственной папке создайте Документ Word с названием «Блок-схема». Произведите установку всех необходимых параметров. Установите панель инструментов **РИСОВАНИЕ**.

Рекомендация: перед тем как приступить к созданию блок-схемы, прощелкайте весь лист клавишей <Enter> - это позволит устанавливать курсор в любую строку листа.

Создание элементов блок-схемы

Для того, чтобы создать блок-схему найдите на панели инструментов кнопку **АВТОФИГУРЫ** Щелкните по ней левой клавишей мыши. Здесь Вы можете выбрать необходимый элемент для Вашей блок-схемы и перенести его на лист. Любая автофигура является объектом редактирования (не забудьте предварительно ее выделить, щелкнув на ней один раз левой клавишей мыши) и на нее распространяются действия по копированию, удалению, всевозможные изменения, содержащиеся в меню **ФОРМАТ АВТОФИГУРЫ**. Все это подробно описано в предыдущей теме. Кроме того, выделенную автофигуру можно просто растягивать или сжимать используя для этого только мышь. Аналогично можно редактировать и стрелки, так как они также являются содержимым меню **АВТОФИГУРЫ**.

В меню **ДЕЙСТВИЯ** Вы можете группировать, разгруппировать, перегруппировать выбранный набор фигур (выбор набора осуществляется при помощи стрелки **ВЫБОР ОБЪЕКТОВ** путем установки и растяжки курсора), установить порядок, сделать сдвиг, выровнять/распределить, повернуть/отразить, изменить автофигуру. Кроме того, возможно осуществление свободного вращения автофигуры, используя соответствующую кнопку

Установка надписи в автофигуру

В автофигуру можно вставить надпись любого содержания. Для этого надо выделить автофигуру, а затем выбрать кнопку **НАДПИСЬ** и щелкнуть изменившимся курсором внутри автофигуры. Теперь вписывайте текст (можете выбрать свой шрифт, интервал и т.д.).

В рамках автофигуры можно изменить направление надписи (в меню **ФОРМАТ строка НАПРАВЛЕНИЕ ТЕКСТА**).

Замечание: если текст не умещается в фигуре, то уменьшите размер выбранного шрифта. Если это не принесло нужного результата, то щелкните по фигуре дважды левой клавишей мыши -появится меню **ФОРМАТ АВТОФИГУРЫ** (ранее это меню мы открывали правой клавишей мыши). Выберите вкладку **НАДПИСЬ** и уберите внутренние поля (просветы) в фигуре, возможно текст поместится.

Дополнительные возможности

Можно изменить цвет автофигуры, установить тень и объем. Все это можно сделать используя меню **ФОРМАТ АВТОФИГУРЫ** и меню **РИСОВАНИЕ** на панели инструментов. Кроме того, установить нужные глубину, освещение, направление, поверхность, цвет объема можно при помощи дополнительной панели (рис. 6, цифра 5), которая вызывается командой **ВИД, ПАНЕЛИ ИНСТРУМЕНТОВ, НАСТРОЙКА, НАСТРОЙКА ОБЪЕМА**.

ЗАДАНИЕ. Создайте блок-схему, изображенную на рис. 4.1.

Для этого выполните следующие действия:

1. В меню **АВТОФИГУРЫ** (или сразу непосредственно на панели инструментов **РИСОВАНИЕ**) выберите овал, начертите его (попробуйте изобразить **овал**, нажимая клавишу <Shift>). Установите глубину 72 пт, направление и освещение любое. Впишите в овал надпись (процесс вписывания см. выше).
2. Начертите стрелку толщиной 0,75 пт.
3. В **АВТОФИГУРАХ** находите и чертите прямоугольник, устанавливайте глубину, направление, руководствуясь рисунком. Затем снова стрелку.
4. Изображая блок условия, нужно сделать надписи над стрелками. Это достигается следующим способом:

- выберите кнопку НАДПИСЬ;
- изобразите прямоугольник на месте будущего слова;
- напечатайте слово, если нужно измените направление текста, шрифт, размер надписи;
- правой клавишей мыши войдите в ФОРМАТ НАДПИСИ и во вкладке ЦВЕТА И ЛИНИИ сделайте установки: НЕТ ЗАЛИВКИ, НЕТ ЛИНИЙ; ОК.

5. Изображайте следующие фигуры блок-схемы, но при этом необходимо учесть следующие замечания:

Замечания:

- для тонкой настройки и подгонки положения фигур (особенно это относится к линиям и стрелкам) используйте стрелки управления курсором с одновременным нажатием клавиши <Ctrl>;
- не старайтесь определить конец линии точно от начала фигуры; удобнее, чтобы линия заходила на фигуру; так как, используя команду ДЕЙСТВИЯ. НА ЗАДНИЙ ПЛАН (предварительно выделив линию), Вы легко достигаете необходимого эффекта;
- при создании блок-схемы следите, чтобы ее части не выходили за выделенные параметры страницы.

6. Завершающий этап

Когда создание блок-схемы завершено, удобно хранить ее в сгруппированном виде. Для этого осуществляется выбор всех объектов блок-схемы при помощи соответствующей клавиши и выполняется команда ДЕЙСТВИЯ, ГРУППИРОВАТЬ. В таком виде удобно копировать созданный рисунок. Кроме того, его можно редактировать (сжимать, растягивать) как единую автофигуру.

1. Понятие алгоритмизации вычислительных процессов

Разработке алгоритма предшествуют такие этапы, как формализация и моделирование задачи. Формализация предполагает замену словесной формулировки решаемой задачи краткими символьными обозначениями, близкими к обозначениям в языках программирования или к математическим. Моделирование задачи является важнейшим этапом, целью которого является поиск общей концепции решения. Обычно моделирование выполняется путем выдвижения гипотез решения задачи и их проверке любым рациональным способом (прикидочные расчеты, физическое моделирование и т.д.). Результатом каждой проверки является либо принятие гипотезы, либо отказ от нее и разработка новой.

При разработке алгоритма используют следующие основные принципы.

Принцип поэтапной детализации алгоритма (другое название - "проектирование сверху-вниз"). Этот принцип предполагает первоначальную разработку алгоритма в виде укрупненных блоков (разбиение задачи на подзадачи) и их постепенную детализацию.

Принцип "от главного к второстепенному", предполагающий составление алгоритма, начиная с главной конструкции. При этом, часто, приходится "достраивать" алгоритм в обратную сторону, например, от середины к началу.

Принцип структурирования, т.е. использования только типовых алгоритмических структур при построении алгоритма. Нетиповой структурой считается, например, циклическая конструкция, содержащая в теле цикла дополнительные выходы из цикла. В программировании нетиповые структуры появляются в результате злоупотребления командой безусловного перехода (GoTo). При этом программа хуже читается и труднее отлаживается.

Говоря о блок-схемах, как о средстве записи алгоритма, можно дать еще один совет по их разработке. Рекомендуется после внесения исправлений в блок-схему аккуратно перерисовывать ее с учетом этих исправлений. Аккуратность записи есть аккуратность мысли программиста. Аккуратно записанный и детализованный алгоритм упрощает его программирование и отладку.

2. Понятие алгоритма и его свойства, способы описания алгоритмов

"Алгоритм" является фундаментальным понятием информатики. Представление о нем необходимо для эффективного применения вычислительной техники к решению практических задач. Алгоритм - это предписание исполнителю (человеку или автомату) выполнить точно определенную последовательность действий, направленных на достижение заданной цели. Алгоритм - это сформулированное на некотором языке правило, указывающее на действия, последовательное выполнение которых приводит от исходных данных к искомому результату. Значение слова алгоритм очень схоже со значением слов рецепт, процесс, метод, способ. Однако любой алгоритм, в отличие от рецепта или способа, обязательно обладает следующими свойствами.

Свойства алгоритма (отличающие его от любых других предписаний): понятность (для конкретного исполнителя); дискретность (команды последовательны, с точной фиксацией моментов начала и конца выполнения команды); точность (после выполнения каждой команды точно известно, завершено ли исполнение алгоритма или же какая команда должна выполняться следующей); результативность (после конечного числа шагов задача решается или же становится ясно, что процесс решения не может быть продолжен); массовость (алгоритм единым образом применяется к любой конкретной формулировке задачи, для которой он разработан).

1. Дискретность - разбиение алгоритма на ряд отдельных законченных действий - шагов. Выполнение алгоритма разбивается на последовательность законченных действий - шагов. Каждое действие должно быть закончено исполнителем алгоритма прежде, чем он приступит к исполнению следующего действия.

2. Точность - однозначные указания. На каждом шаге однозначно определено преобразование объектов среды исполнителя, полученной на предыдущих шагах алгоритма. Если алгоритм многократно применяется к одному и тому же набору исходных данных, то на выходе он получает каждый раз один и тот же результат. Запись алгоритма должна быть такой, чтобы на каждом шаге его выполнения было известно, какую команду надо выполнять следующей.

3. Понятность - однозначное понимание и исполнение каждого шага алгоритма его исполнителем. Алгоритм должен быть записан на понятном для исполнителя языке.

4. Результативность - обязательное получение результата за конечное число шагов. Каждый шаг (и алгоритм в целом) после своего завершения дает среду, в которой все объекты однозначно определены. Если это по каким-либо причинам невозможно, то алгоритм должен сообщать, что решение задачи не существует. Работа алгоритма должна быть завершена за конечное число шагов. Информатика оперирует только с конечными объектами и конечными процессами, поэтому вопрос о рассмотрении бесконечных алгоритмов остается за рамками теории алгоритмов.

5. Массовость - применение алгоритма к решению целого класса однотипных задач.

Порядок выполнения алгоритма:

- Действия в алгоритме выполняются в порядке их записи
- Нельзя менять местами никакие два действия алгоритма
- Нельзя не закончив одного действия переходить к следующему

Для записи алгоритмов используются специальные языки:

1. Естественный язык (словесная запись)
2. Формулы
3. Псевдокод
4. Структурограммы
5. Синтаксические диаграммы
6. Графический (язык блок-схем)

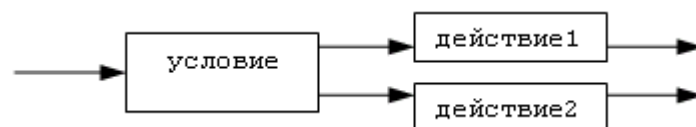
1. Естественный язык:

если условие то действие1 иначе действие2

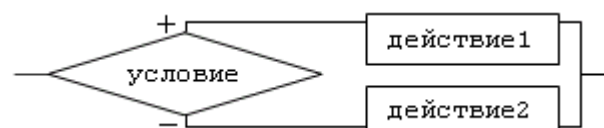
2. Структурограмма:



3. Синтаксическая диаграмма:

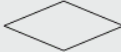
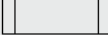


4. Графический язык:



Составление алгоритмов графическим способом подчиняется двум ГОСТам:

1. ГОСТ 19.002-80, соответствует международному стандарту ИСО 2636-73. Регламентирует правила составления блок-схем.
2. ГОСТ 19.003-80, соответствует международному стандарту ИСО 1028-73. Регламентирует использование графических примитивов.

Название	Символ (рисунок)	Выполняемая функция (пояснение)
1. Блок вычислений		Выполняет вычислительное действие или группу действий
2. Логический блок		Выбор направления выполнения алгоритма в зависимости от условия
3. Блоки ввода/вывода		Ввод или вывод данных вне зависимости от физического носителя
		Вывод данных на печатающее устройство
4. Начало/конец (вход/выход)		Начало или конец программы, вход или выход в подпрограмму
5. Предопределенный процесс		Вычисления по стандартной или пользовательской подпрограмме
6. Блок модификации		Выполнение действий, изменяющих пункты алгоритма
7. Соединитель		Указание связи между прерванными линиями в пределах одной страницы
8. Межстраничный соединитель		Указание связи между частями схемы, расположенной на разных страницах

Правила построения блок-схем:

Блок-схема выстраивается в одном направлении либо сверху вниз, либо слева направо

Все повороты соединительных линий выполняются под углом 90 градусов

i. Задания к занятию

Построить алгоритмы решения задач

1. Даны три действительных числа a , b , c . Вывести на экран те, которые принадлежат интервалу (x, y) .
2. Даны два действительных числа x и y . Если x и y отрицательные, то каждое значение заменить его модулем; если отрицательное только одно из них, то x и y увеличить на 0.5; если оба неотрицательные, то x и y увеличить в 10 раз.
3. Известны время старта и время финиша двух лыжников. Если один из лыжников выигрывает, то он получает поощрительный приз и слова поздравления. А проигравший лыжник ничего не выигрывает.
4. В книжном магазине продаж газет занимается ЭВМ. Составить программу, запрашивающую стоимость товара и сумму денег, внесенных покупателем. Определить причитающую сдачу, или выдать сообщение, что денег внесено недостаточно или достаточно.

5. Составить программу, запрашивающую имя и год рождения пользователя. ЭВМ должна выдать сообщение, имеете ли вы право голоса на выборы. Если нет, то выдать сообщение, через сколько лет вы это право получите.
6. Даны два действительных числа a и b . Если числа не равны, то заменить каждое из них числами равными $(a+b)$ и $(a-b)$ соответственно, а если равны, то заменить их нулями.

Протабулировать функции:

7. $K = a * b^3 * c + \frac{\sqrt{c-4}}{b+4}$, где c изменяется на заданном интервале с заданным шагом, a, b – любые значения.

8. $R = X^2 - \sqrt{b+2*c} + \frac{1}{c}$, где x изменяется на заданном интервале с заданным шагом, c, b – любые значения.

9. $Z = a * X^a + \cos \sqrt{(x+b)}$, где x изменяется на заданном интервале с заданным шагом, a, b – любые значения.

10. $W = a * b * c^2 + \frac{\sqrt{a-2}}{c}$, где a изменяется на заданном интервале с заданным шагом, c, b – принимают одно любое значение.

11. $T = \sqrt{X^2 - 12} + \frac{y-x}{2*y}$, где x изменяется на заданном интервале с заданным шагом, y – одно любое значение.

12. $Q = 10 * a + \sqrt[3]{b-c} - \frac{1}{c}$, где a изменяется на заданном интервале с заданным шагом, c, b – любые значения.

13.
$$M = \begin{cases} \sqrt{x} + a * X^3, & \text{если } x > 0 \\ \frac{a * X^2 + 1}{a - 2}, & \text{если } x \leq 0 \end{cases}$$

при $x = \sqrt{y+4} - 10$, где y изменяется на заданном интервале с заданным шагом, a – принимает одно любое значение, $x = 9.2$.

14.
$$B = \begin{cases} 13.7 * a^3 + z * \sqrt{a+z}, & \text{если } z \geq 0 \\ \frac{1}{\sqrt{a - \sqrt[3]{z}}} + z^2, & \text{если } z < 0 \end{cases}$$

при $z=x+6$, где x изменяется на заданном интервале с заданным шагом, a – любые значения.

$$15. \quad Z = \begin{cases} \frac{4 * d \sqrt{b}}{7 * a + d^2} - x, & \text{если } x \geq 0 \\ a + x * d - 1/b, & \text{если } x < 0 \end{cases}$$

при $x=(y-2)^2$, где y изменяется на заданном интервале с заданным шагом, a, b, d – любые значения.

$$16. \quad Y = \begin{cases} a * b + 1.4 * z, & \text{если } z = 0 \\ z + \frac{b * \sqrt{a+2}}{1+z^2}, & \text{если } z > 0 \\ \sqrt{a+2} - 10 * a + 1/b, & \text{если } z < 0 \end{cases}$$

при $z=2*x-4$, где x изменяется на заданном интервале с заданным шагом, a, b – любые значения.

$$17. \quad D = \begin{cases} \sqrt{b} + a * x^3, & \text{если } x > 0 \\ \frac{b * a^2}{a+2} - x, & \text{если } x = 0 \\ x - \sqrt{x-a}, & \text{если } x < 0 \end{cases}$$

при $x=10-4*y$, где y изменяется на заданном интервале с заданным шагом, a, b – любые значения.

$$18. \quad Q = \begin{cases} 10 * a * b + 1/z, & \text{если } a \geq 0 \\ \frac{1}{a^2 * b} + \sqrt[3]{z}, & \text{если } a < 0 \end{cases}$$

при $a=5-2*x$, где x изменяется на заданном интервале с заданным шагом, b, z – любые значения.

$$19. \quad P = \begin{cases} x - 2 * y + 4, & \text{если } x \leq 0 \\ 1/y + 2 * y + x/3, & \text{если } x > 0 \end{cases}$$

при $x=\sqrt{2+z}-4$, где z изменяется на заданном интервале с заданным шагом, y – одно любое значение.

$$20. \quad B = \begin{cases} \frac{b * X^2}{a + 2} + 15, & \text{если } x \leq 3 \\ \sqrt{x - a} + a * b, & \text{если } x > 3 \end{cases}$$

при $x = 2 * y + 3$, где y изменяется на заданном интервале с заданным шагом, a, b – любые значения.

$$21. \quad S = \begin{cases} \frac{1}{a * X^4} + b^3, & \text{если } x < 3 \\ \sqrt[3]{\sqrt{2 * a - b}} - 4, & \text{если } x \geq 3 \end{cases}$$

при $x = \frac{1}{y} + 9$, где y изменяется на заданном интервале с заданным шагом, a, b – любые значения.

$$22. \quad K = \begin{cases} 6 * a^3 + \frac{2 * a - b}{b}, & \text{если } t \leq 0.1 \\ t^2 - \sqrt{a + b}, & \text{если } t > 0.1 \end{cases}$$

при $t = \sqrt{x} + 4$, где x изменяется на заданном интервале с заданным шагом, a, b – любые значения.

$$23. \quad R = \begin{cases} x * y * Z^2 + \sqrt{x + y}, & \text{если } z > 0 \\ \frac{2 + z}{\sqrt[3]{x * y - z}} - \frac{4 * x - y}{3}, & \text{если } z \leq 0 \end{cases}$$

где Z изменяется на заданном интервале с заданным шагом, x – любые значения, y – одно любое значение

24. Дано количество студентов. Ввести имена студентов и их год рождения. Определить, через сколько лет они получат право голоса или сделать вывод, что они имеют право голоса.

25. Рассчитать $y = \frac{a * b + 2}{4 * b}$ для известного количества любых значений a и b .

26. Рассчитать $S = \sqrt{2 * a - 3 * b} + 4$ для известного количества любых значений a и b .

27. Рассчитать $Q = \frac{b * X^2}{\ell^x - 1}$ для известного количества любых значений x и b .
28. Рассчитать все возможные значения $x = \frac{2 * a - b^3}{2}$, если a изменяется на заданном интервале с заданным шагом, b изменяется на заданном интервале с заданным шагом.
29. Рассчитать все возможные значения $f = \frac{(a+b)^2 - 4}{2}$, если a изменяется на заданном интервале с заданным шагом, b изменяется на заданном интервале с заданным шагом.
30. Рассчитать все возможные значения $k = \frac{4 * x - (y - 2)}{3}$, если x изменяется на заданном интервале с заданным шагом, y принадлежит известное любое количество значений.
31. Рассчитать все возможные значения $p = \frac{(a+b)^2 - 12}{3}$, если a изменяется на заданном интервале с заданным шагом, b изменяется на заданном интервале с заданным шагом.
32. Сохранить в памяти ЭВМ индексы элементов массива $C(N)$, значения которых попадают в интервал от $A1$ до $A2$.
33. Рассчитать $T = \sqrt{X^2 - 12} + \frac{y - x}{2 * y}$ для известного количества любых значений y и x . Сохранить в памяти ЭВМ только положительные значения T .
34. Рассчитать $X = a^2 - \sqrt{b + 4} - \frac{b}{a}$ для известного количества любых значений a и b . Сохранить в памяти ЭВМ только отрицательные значения X .
35. Дано: список фирм и их уставной фонд. Найти сумму уставных фондов фирм, стоящих на нечетных местах.
36. Дано: список фирм и их уставной фонд. Найти сумму фондов, попадающих в интервал от $B1$ до $B2$.

37. Дано: список фирм и их уставной фонд. Подсчитать количество фирм, уставной фонд которых выше K , или вывести сообщение, что таких фирм нет.
38. Подсчитать сумму и количество элементов, попадающих в интервал от V_1 до V_2 в массиве $A(N)$.
39. Даны массивы $A(N)$ и $B(N)$. Найти разность значений A и B и переписать в массив C . В массиве $C(N)$ упорядочить элементы по возрастанию.
40. Дано: список фирм и их уставной фонд. Вывести название 3 фирм с наибольшим уставным фондом.
41. Удалить первый отрицательный элемент в массиве $B(K)$ или сделать вывод, что отрицательных элементов в массиве нет.
42. В массиве $B(V)$ исключить все элементы с отрицательными значениями.
43. Вычислить сумму положительных элементов каждой строки матрицы $A(M,N)$.
44. Переписать элементы двумерного массива в одномерный.
45. Вычислить сумму элементов главной диагонали матрицы $B(K,L)$.
46. Задан массив $A(M,N)$. Сформировать новый массив B по формуле: $B(I,J)=I*J*A(I,J)$.
47. Определить количество положительных и отрицательных элементов матрицы $A(K,L)$.
48. Сформировать одномерный массив B из положительных элементов главной диагонали матрицы $A(M,N)$.
49. Вычислить сумму отрицательных элементов по второй диагонали матрицы $C(Q,Z)$.
50. Найти наибольший элемент матрицы $V(N,M)$ и номер строки и столбца, на пересечении которых он стоит.
51. Сформировать одномерный массив из элементов матрицы $D(K,L)$, попадающих в промежуток от A до B .
52. Определить, есть ли в массиве $M(A,B)$ отрицательный элемент.
53. Определить, есть ли в массиве $K(L,A)$ строка, состоящая только из положительных элементов.
54. Определить, есть ли в массиве $S(K,L)$ столбец, состоящий только из отрицательных элементов.
55. Определить, есть ли в массиве $T(A,B)$ столбец, состоящий только из элементов, принадлежащих промежутку от K до L .
56. В матрице $X(A,B)$ все ненулевые элементы заменить противоположными по знаку.
57. Матрица $K(M,N)$ состоит из единиц и нулей. Найти в ней номера (индексы) хотя бы одной строки, не содержащих единицы, либо сообщить, что таковых нет.

58. Матрица $K(M,N)$ состоит из единиц и нулей. Найти в ней номера (индексы) хотя бы одного столбца, не содержащих нулей, либо сообщить, что таковых нет.
59. Найти наибольшие элементы каждого столбца матрицы $M(A,B)$ и записать их в массив N .
60. Поменять местами строки массива, содержащие $\max$ и $\min$ элементы массива $B(K,L)$. Если эти элементы находятся в одной строке – выдать об этом сообщение.