МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН

ТЕРМЕЗСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Лекции

по курсу

«ИНФОРМАТИКА»

«ИНФОРМАТИКА»

для направления бакалавриата
5220100 – «Русский язык»
ВЫСШЕГО ОБРАЗОВАНИЯ

Термез- 2006

«Утверждаю»
Рассмотрено и одобрено
На заседании Совета факультета
«Физико-математики»
Председатель Совета_____проф.Ш. Жураев
« ___» _____2006 г.

«Рекомендовано»
Обсуждено и рекомендовано
на заседании кафедры
«Информатика и прикладная математика»
Зав. кафедрой _____доц. Ч.Б. Нормуродов
(Протокол № ____заседания кафедры
от «____» _____2006 г.)

Составитель:
М.Дж. Зарипова, дисциплина «Информатика»

(рабочая программа). – Т.: ТГУ, 2006 г.

Оглавление

Тема 1. Введение в информатику. Характеристика и сущность предмета Информатика.

План:

1. Характеристика и сущность предмета Информатика.
2. Что такое информация?
3. Обработка, накопление и передача информации.
4. Какими свойствами обладает информация?
5. Что понимают под информатизацией общества?

Что такое информатика?

Термин *"информатика"* (франц. ***informatique***) происходит от французских слов ***information*** (информация) и ***automatique*** (автоматика) и дословно означает *"информационная автоматика"*.

Широко распространён также англоязычный вариант этого термина — *"Computer science"*, что означает буквально *"компьютерная наука"*.

Информатика — это основанная на использовании компьютерной техники дисциплина, изучающая структуру и общие свойства информации, а также закономерности и методы её создания, хранения, поиска, преобразования, передачи и применения в различных сферах человеческой деятельности.

В 1978 году международный научный конгресс официально закрепил за понятием *"информатика"* области, связанные с разработкой, созданием, использованием и материально-техническим обслуживанием систем обработки информации, включая компьютеры и их программное обеспечение, а также организационные, коммерческие, административные и социально-политические аспекты компьютеризации — массового внедрения компьютерной техники во все области жизни людей.

Таким образом, информатика базируется на компьютерной технике и немыслима без нее.

Информатика — научная дисциплина с широчайшим диапазоном применения. Её основные направления:

- разработка вычислительных систем и программного обеспечения;
- теория информации, изучающая процессы, связанные с передачей, приёмом, преобразованием и хранением информации;
- методы искусственного интеллекта, позволяющие создавать программы для решения задач, требующих определённых интеллектуальных усилий при выполнении их человеком (логический вывод, обучение, понимание речи, визуальное восприятие, игры и др.);
- системный анализ, заключающийся в анализе назначения проектируемой системы и в установлении требований, которым она должна отвечать;
- методы машинной графики, анимации, средства мультимедиа;
- средства телекоммуникации, в том числе, глобальные компьютерные сети, объединяющие всё человечество в единое информационное сообщество;
- разнообразные приложения, охватывающие производство, науку, образование, медицину, торговлю, сельское хозяйство и все другие виды хозяйственной и общественной деятельности.

Информатика - это новая научная дисциплина и новая информационная индустрия, связанные с использованием персональных компьютеров и сетей ЭВМ. В новом тысячелетии предполагается, что основная информация, связанная с деятельностью людей будет храниться в памяти электронных вычислительных машин.

Информатика как **научная дисциплина** изучает законы, принципы и методы накопления, обработки и передачи информации с помощью ЭВМ. В этом смысле информатика как наука является фундаментом для развития новой информационной индустрии, основанной на использовании сетей ЭВМ.

Фундамент информатики образуют вычислительные науки - науки об вычислительных процессах и организации вычислительных машин, вычислительных систем и сетей. Основным объектом вычислительных наук являются вычислительные машины - устройства для организации вычислений и обработки символьной информации.

Информатику обычно представляют состоящей из двух частей:

- технические средства;
- программные средства.

Технические средства, то есть *аппаратура компьютеров*, в английском языке обозначаются словом ***Hardware***, которое буквально переводится как *"твёрдые изделия"*.

А для *программных средств* выбрано (а точнее, создано) очень удачное слово ***Software*** (буквально — *"мягкие изделия"*), которое подчёркивает равнозначность программного обеспечения и самой машины и вместе с тем подчёркивает способность программного обеспечения модифицироваться, приспосабливаться, развиваться.

Программное обеспечение — это совокупность всех программ, используемых компьютерами, а также вся область деятельности по их созданию и применению.

Помимо этих двух общепринятых ветвей информатики выделяют ещё одну существенную ветвь — *алгоритмические средства*. Для неё российский академик А.А. Дородницин предложил название ***Brainware*** (от англ. ***brain*** — интеллект). Эта ветвь связана с разработкой алгоритмов и изучением методов и приёмов их построения.

Алгоритмы — это правила, предписывающие выполнение последовательностей действий, приводящих к решению задачи.

Нельзя приступить к программированию, не разработав предварительно алгоритм решения задачи.

Роль информатики в развитии общества чрезвычайно велика. С ней связано начало революции в области накопления, передачи и обработки информации. Эта революция, следующая за революциями в овладении веществом и энергией, затрагивает и коренным образом преобразует не только сферу материального производства, но и интеллектуальную, духовную сферы жизни.

Рост производства компьютерной техники, развитие информационных сетей, создание новых информационных технологий приводят к значительным изменениям во всех сферах общества: в производстве, науке, образовании, медицине и т.д.

Что такое информация?

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

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

Обработка, накопление и передача информации происходит не только внутри ЭВМ. Передачу и накопление информации мы видим при общении людей, в технических устройствах, в живых организмах и в жизни общества, что тоже входит в предмет изучения информатики как научной дисциплины.

Передача информации в общении людей - это передача сведений и суждений, данных и сообщений. Даже улыбка является передачей информации при общении людей друг с другом. Любая совместная деятельность людей - работа, учеба и даже игра - построены на обмене и передаче информации.

Для **живых существ** восприятие и передача информации в форме сигналов - основное отличие от неодушевленных предметов окружающего мира. Языковая форма передачи знаковой информации - основное отличие людей от других живых существ.

Слово **информация** происходит от латинского **informatio,** означающего сведения, разъяснения, пояснения. С содержательной точки зрения информация - это сведения о ком-то или о чем-то, а с формальной точки зрения - набор знаков и сигналов.

С юридической точки зрения информация - это сведения о людях, предметах, фактах, событиях и процессах, независимо от формы их представления.

В обиходе информацией называют любые данные или сведения, которые кого-либо интересуют.

Например, сообщение о каких-либо событиях, о чьей-либо деятельности и т.п. *"Информировать"* в этом смысле означает *"сообщить нечто, неизвестное раньше"*.

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

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

Так, сообщение, составленное на японском языке, не несёт никакой новой информации человеку, не знающему этого языка, но может быть высокоинформативным для человека, владеющего японским. Никакой новой информации не содержит и сообщение, изложенное на знакомом языке, если его содержание непонятно или уже известно.

Информация есть характеристика не сообщения, а *соотношения между сообщением и его потребителем*. Без наличия потребителя, хотя бы потенциального, говорить об информации бессмысленно.

В случаях, когда говорят об автоматизированной работе с информацией посредством каких-либо технических устройств, обычно в первую очередь интересуются не содержанием сообщения, а тем, сколько символов это сообщение содержит.

Применительно к компьютерной обработке данных под информацией понимают некоторую последовательность символических обозначений (букв, цифр, закодированных графических образов и звуков и т.п.), несущую смысловую нагрузку и представленную в понятном компьютеру виде. Каждый новый символ в такой последовательности символов увеличивает информационный объём сообщения.

В каком виде существует информация?

Информация может существовать в самых разнообразных формах:

- в виде текстов, рисунков, чертежей, фотографий;
- в виде световых или звуковых сигналов;
- в виде радиоволн;
- в виде электрических и нервных импульсов;
- в виде магнитных записей;
- в виде жестов и мимики;
- в виде запахов и вкусовых ощущений;
- в виде хромосом, посредством которых передаются по наследству признаки и свойства организмов и т.д.

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

Как передаётся информация?

Информация передаётся в виде сообщений от некоторого источника информации к её приёмнику посредством канала связи между ними. Источник посылает передаваемое сообщение, которое кодируется в передаваемый сигнал. Этот сигнал посылается по каналу связи. В результате в приёмнике появляется принимаемый сигнал, который декодируется и становится принимаемым сообщением.

Примеры:

1. *сообщение, содержащее информацию о прогнозе погоды, передаётся приёмнику* (телезрителю) *от источника* — специалиста-метеоролога *посредством канала связи* — телевизионной передающей аппаратуры и телевизора;
2. *живое существо своими органами чувств* (глаз, ухо, кожа, язык и т.д.) *воспринимает информацию из внешнего мира, перерабатывает её* в определенную последовательность нервных импульсов, *передает* импульсы по нервным волокнам, *хранит* в памяти в виде состояния нейронных структур мозга, *воспроизводит* в виде звуковых сигналов, движений и т.п., *использует* в процессе своей жизнедеятельности.

Передача информации по каналам связи часто сопровождается воздействием помех, вызывающих искажение и потерю информации.

Что можно делать с информацией?

Информацию можно:

▪ создавать; ▪ передавать; ▪ воспринимать; ▪ использовать; ▪ запоминать; ▪ принимать; ▪ копировать;	▪ формализовать; ▪ распространять; ▪ преобразовывать; ▪ комбинировать; ▪ обрабатывать; ▪ делить на части; ▪ упрощать;	▪ собирать; ▪ хранить; ▪ искать; ▪ измерять; ▪ разрушать; ▪ и др.

Все эти процессы, связанные с определенными операциями над информацией, называются информационными процессами.

Какими свойствами обладает информация?

Свойства информации:

▪ достоверность; ▪ полнота; ▪ ценность; ▪ своевременность;	▪ понятность; ▪ доступность; ▪ краткость; ▪ и др.

Информация достоверна, если она отражает истинное положение дел. Недостоверная информация может привести к неправильному пониманию или принятию неправильных решений.

Достоверная информация со временем может стать недостоверной, так как она обладает свойством устаревать, то есть перестаёт отражать истинное положение дел.

Информация полна, если её достаточно для понимания и принятия решений. Как неполная, так и избыточная информация сдерживает принятие решений или может повлечь ошибки.

Точность информации определяется степенью ее близости к реальному состоянию объекта, процесса, явления и т.п.

Ценность информации зависит от того, насколько она важна для решения задачи, а также от того, насколько в дальнейшем она найдёт применение в каких-либо видах деятельности человека.

Только своевременно полученная информация может принести ожидаемую пользу. Одинаково нежелательны как преждевременная подача информации (когда она ещё не может быть усвоена), так и её задержка.

Если ценная и своевременная информация выражена непонятным образом, она может стать бесполезной.

Информация становится понятной, если она выражена языком, на котором говорят те, кому предназначена эта информация.

Информация должна преподноситься в доступной (по уровню восприятия) форме. Поэтому одни и те же вопросы по разному излагаются в школьных учебниках и научных изданиях.

Информацию по одному и тому же вопросу можно изложить кратко (сжато, без несущественных деталей) или пространно (подробно, многословно). Краткость информации необходима в справочниках, энциклопедиях, учебниках, всевозможных инструкциях.

Что такое обработка информации?

Обработка информации – получение одних информационных объектов из других информационных объектов путем выполнения некоторых алгоритмов

Обработка является одной из основных операций, выполняемых над информацией, и главным средством увеличения объёма и разнообразия информации.

Средства обработки информации — это всевозможные устройства и системы, созданные человечеством, и в первую очередь, компьютер — универсальная машина для обработки информации.

Компьютеры обрабатывают информацию путем выполнения некоторых алгоритмов.

Живые организмы и растения обрабатывают информацию с помощью своих органов и систем.

Особую роль для общества играет документированная информация. Документы - это информация, зафиксированная на материальном носителе - бумаге или машинном носителе, имеющем реквизиты, позволяющие его идентифицировать.

Возможность записи информации в письменном виде - в форме последовательности знаков - привела к образованию государств, возникновению бюрократии и появлению почтовых служб. Параллельно это привело - к появлению грамотных людей - людей, умеющих читать, писать и искать информацию для решения различных проблем.

Возникновение письменности позволило людям не только передавать информацию, но и накапливать ее в форме записок, писем и рукописей в архивах, а также в личных и публичных

библиотеках. Квалифицированная переработка информации потребовала людей, имеющих надлежащее образование.

Для обучения грамотности были открыты гимназии, лицеи и школы, а для подготовки образованных людей - университеты и колледжи, где накоплением и передачей знаний стали заниматься ученые, учителя и профессора.

Для **хранения знаний** стали использоваться рукописные книги, а для хранения книг - библиотеки и книгохранилища. В них стал накапливаться интеллектуальный потенциал общества и государств. Неслучайно одной из основных задач варвары считали уничтожение книг и книгохранилищ.

Изобретение **печатных станков** в XV в. создало технологическую основу для массового издания и распространения печатных книг. Это послужило основой для всеобщего распространения грамотности и открытия массовых начальных школ, в которых все дети обучались грамотности - умениям читать, писать и считать.

Основные термины

Информатика, информация, передача информации, обработка информации, свойства информации.

К о н т р о л ь н ы е в о п р о с ы

1. Что такое информация?
2. Что такое информатика?
3. В каком виде существует информация?
4. Как передаётся информация?
5. Что можно делать с информацией?
6. Какими свойствами обладает информация?
7. Что такое обработка информации?
8. Где будет храниться информация в XXI веке?

Тема 2. История развития ЭВМ. Классификация по поколениям.

План:

1. Компьютеры первого поколения.
2. Компьютеры второго поколения.
3. Компьютеры третьего поколения.
4. Компьютеры четвертого поколение.
5. Компьютеры пятого поколение.
6. Развитие вычислительной техники.

Развитие промышленного производства в XVIII-XIX веках потребовало большого числа специалистов, для подготовки которых было открыто большое число университетов. Это дало мощный толчок для развития естественных наук - химии, физики, механики, математики и подготовки инженерных кадров.

Развитие печатных станков привело к **появлению и распространению газет** как средств массовой информации и информатизации общества, а также появлению и распространению журналов для распространения литературных произведений. В это же время появились первые законы, регулирующие авторские права.

Изобретение в XIX - начале XX века **телеграфа, радио и телефона** открыло новые возможности в передаче информации и информатизации общества. Эти технические средства дали возможность практически мгновенно передавать информацию на любые расстояния.

Следующим шагом технического прогресса стало появление и развитие **электроники, телевидения и радиовещания** к середине XX века. Изобретение телевидения позволило людям видеть на экранах телевизоров события, происходящие в самых различных точках планеты, а изобретение магнитофона - накапливать звуковую и видеоинформацию на магнитных носителях.

Точкой отсчета становления информатики как индустрии стало изобретение в середине XX века **электронных вычислительных машин.** Основной особенностью компьютеров стала возможность автоматической обработки информации. Переработка информации перестала быть исключительной способностью людей и живых существ.

Параллельно в середине XX века были заложены **теоретические основы информатики** как научной дисциплины. В этот период получили развитие математич-еская логика - фундамент теоретической информатики и теория алгоритмов-фундамент вычислительных наук.

На чем основана классификация по поколениям?

Деление компьютерной техники на поколения — весьма условная, нестрогая классификация вычислительных систем по степени развития аппаратных и программных средств, а также способов общения с компьютером.

Идея делить машины на поколения вызвана к жизни тем, что за время короткой истории своего развития компьютерная техника проделала большую эволюцию как в смысле элементной базы (*лампы*, *транзисторы*, *микросхемы* и др.), так и в смысле изменения её структуры, появления новых возможностей, расширения областей применения и характера использования.

I поколение, 1945-1954 гг

Какие компьютеры относятся в первому поколению?

К первому поколению обычно относят машины, созданные на рубеже 50-х годов. В их схемах использовались электронные лампы. Эти компьютеры были огромными, неудобными и слишком дорогими машинами, которые могли приобрести только крупные корпорации и правительства. Лампы потребляли огромное количество электроэнергии и выделяли много тепла. Электронная лампа Компьютер "Эниак".

Первое поколение

Набор команд был небольшой, схема арифметико-логического устройства и устройства управления достаточно проста, программное обеспечение практически отсутствовало. Показатели объема оперативной памяти и быстродействия были низкими. Для ввода-вывода использовались перфоленты, перфокарты, магнитные ленты и печатающие устройства. Быстродействие порядка 10-20 тысяч операций в секунду. Но это только техническая сторона. Очень важна и другая — способы использования компьютеров, стиль программирования, особенности математического обеспечения.

Программы для этих машин писались на языке конкретной машины. Математик, составивший программу, садился за пульт управления машины, вводил и отлаживал программы и производил по ним счет. Процесс отладки был наиболее длительным по времени.

Несмотря на ограниченность возможностей, эти машины позволили выполнить сложнейшие расчёты, необходимые для прогнозирования погоды, решения задач атомной энергетики и др. Опыт использования машин первого поколения показал, что существует огромный разрыв между временем, затрачиваемым на разработку программ, и временем счета.

ЭВМ "Урал" Эти проблемы начали преодолевать путем интенсивной разработки средств автоматизации программирования, создания систем обслуживающих программ, упрощающих работу на машине и увеличивающих эффективность её использования. Это, в свою очередь, потребовало значительных изменений в структуре компьютеров, направленных на то, чтобы приблизить её к требованиям, возникшим из опыта эксплуатации компьютеров. Отечественные машины первого поколения: МЭСМ (малая электронная счётная машина), БЭСМ, Стрела, Урал, М-20.

Применение вакуумно-ламповой технологии, использование систем памяти на ртутных линиях задержки, магнитных барабанах, электронно-лучевых трубках (трубках Вильямса). Для ввода-вывода данных использовались перфоленты и перфокарты, магнитные ленты и печатающие устройства. Была реализована концепция хранимой программы.

Компьютеры **первого поколения** создавались именно как электронные вычислительные машины для автоматизации сложнейших вычислений оборонного и научного характера. Объем и сложность вычислений, выполнявшихся первыми компьютерами, были недоступны даже самым сильным математикам и вычислителям, но посильными для современных домашних компьютеров.

В этот период появились первые **профессиональные программисты** и первые теоретические работы по математической лингвистике, теории искусственного интеллекта и теоретическому программированию. Бурное развитие получили вычислительная и дискретная математика, образующие математическую базу информатики и вычислительных наук.

II поколение, 1955-1965 гг

Какие компьютеры относятся ко второму поколению?

Второе поколение компьютерной техники — машины, сконструированные примерно в 1955-65 гг. Характеризуются использованием в них как электронных ламп, так и дискретных транзисторных логических элементов. Их оперативная память была построена на магнитных сердечниках. В это время стал расширяться диапазон применяемого оборудования ввода-вывода, появились высокопроизводительные устройства для работы с магнитными лентами, магнитные барабаны и первые магнитные диски. Память на магнитных сердечниках. Быстродействие — до сотен тысяч операций в секунду, ёмкость памяти — до нескольких десятков тысяч слов.

Появились так называемые языки высокого уровня, средства которых допускают описание всей необходимой последовательности вычислительных действий в наглядном, легко воспринимаемом виде.

Программа, написанная на алгоритмическом языке, непонятна компьютеру, воспринимающему только язык своих собственных команд. Поэтому специальные программы, которые называются трансляторами, переводят программу с языка высокого уровня на машинный язык.

Появился широкий набор библиотечных программ для решения разнообразных математических задач. Появились мониторные системы, управляющие режимом трансляции и исполнения программ. Из мониторных систем в дальнейшем выросли современные операционные системы.

Операционная система — важнейшая часть программного обеспечения компьютера, предназначенная для автоматизации планирования и организации процесса обработки программ, ввода-вывода и управления данными, распределения ресурсов, подготовки и отладки программ, других вспомогательных операций обслуживания. Таким образом, *операционная система является программным расширением устройства управления компьютера*.

Для некоторых машин второго поколения уже были созданы операционные системы с ограниченными возможностями. Машинам второго поколения была свойственна программная несовместимость, которая затрудняла организацию крупных информационных систем. Поэтому в середине 60-х годов наметился переход к созданию компьютеров, программно совместимых и построенных на микроэлектронной технологической базе.

Замена электронных ламп как основных компонентов компьютера на транзисторы. Компьютеры стали более надежными, быстродействие их повысилось, потребление энергии уменьшилось. С появлением памяти на магнитных сердечниках цикл ее работы уменьшился до десятков микросекунд. Главный принцип структуры - централизация. Появились высокопроизводительные устройства для работы с магнитными лентами, устройства памяти на магнитных дисках.

Компьютеры **второго поколения** создавались в качестве универсальных вычислительных машин, предназначенных для решения задач обработки и накопления информации с использованием устройств ввода и вывода. Компьютеры этого поколения стали использоваться для решения различных научных, экономических, оборонных и инженерных задач.

Для этих машин были созданы **первые операционные системы,** системы программирования и первые диалоговые системы. В этот период программирование зародилось как профессия и появились первые языки программирования и первые инструментальные программы - компиляторы и интерпретаторы для ЭВМ.

III поколение, 1965-1974 гг.

В чем особенности компьютеров третьего поколения?

Компьютер IBM-360. Третье поколение

Машины третьего поколения созданы примерно после 60-х годов. Поскольку процесс создания компьютерной техники шел непрерывно, и в нём участвовало множество людей из разных стран, имеющих дело с решением различных проблем, трудно и бесполезно пытаться установить, когда "поколение" начиналось и заканчивалось. Возможно, наиболее важным критерием различия машин второго и третьего поколений является критерий, основанный на понятии архитектуры.

Интегральная схема. Машины третьего поколения — это семейства машин с единой архитектурой, т.е. программно совместимых. В качестве элементной базы в них используются интегральные схемы, которые также называются микросхемами.

Машины третьего поколения имеют развитые операционные системы. Они обладают возможностями мультипрограммирования, т.е. одновременного выполнения нескольких программ. Многие задачи управления памятью, устройствами и ресурсами стала брать на себя операционная система или же непосредственно сама машина.

Примеры машин третьего поколения — семейства IBM-360, IBM-370, ЕС ЭВМ (Единая система ЭВМ), СМ ЭВМ (Семейство малых ЭВМ) и др.

Быстродействие машин внутри семейства изменяется от нескольких десятков тысяч до миллионов операций в секунду. Ёмкость оперативной памяти достигает нескольких сотен тысяч слов.

Краткое описание процесса изготовления микросхем.

1. Разработчики с помощью компьютера создают электрическую схему новой микросхемы. Для этого они вводят в компьютер перечень свойств, которыми должна обладать микросхема, а компьютер с помощью специальной программы разрабатывает детальную структуру соединений и конструкций всех взаимодействующих элементов микросхемы.
2. Компьютер создаёт схемы расположения элементов на поверхности полупроводникового кристалла кремния. По этим схемам изготавливаются фотошаблоны — стеклянные пластинки со штриховым рисунком. Через фотошаблоны специальными лампами или источниками рентгеновского излучения, а иногда, и электронными пучками, освещают (засвечивают) нанесённый на поверхность кристалла кремния слой фото- или, соответственно, рентгеночувствительного лака.
3. Засвеченные (или, наоборот, незасвеченные) участки лака меняют свои свойства и удаляются специальными растворителями. Этот процесс называется травлением. Вместе с лаком с поверхности кристалла кремния удаляется и слой окисла, и эти места становятся доступными для легирования — внедрения в кристаллическую решётку кремния атомов бора или фосфора. Легирование обычно требует нагрева пластинки в парах нужного элемента до 1100 - 1200 °С.
4. Последовательно меняя шаблоны и повторяя процедуры травления и легирования, создают один за другим слои будущей микросхемы. При этом на одной пластинке кристалла кремния создаётся множество одинаковых микросхем.
5. Каждая микросхема проверяется на работоспособность. Негодные выбраковываются.

После завершения всех операций пластинки разрезаются на отдельные кристаллики с микросхемами, к ним присоединяют выводы и устанавливают в корпуса.

Компьютеры проектировались на основе интегральных схем малой степени интеграции (МИС - 10 - 100 компонентов на кристалл) и средней степени интеграции (СИС - 10 -1000 компонентов на кристалл).

Появилась идея, которая и была реализована, проектирования семейства компьютеров с одной и той же архитектурой, в основу которой положено главным образом программное обеспечение. В конце 60-х появились мини-компьютеры. В 1971 году появился первый микропроцессор.

Третье поколение компьютеров - это первые серийные вычислительные машины для автоматизации обработки и накопления информации. Для этих ЭВМ был создан целый спектр устройств ввода, вывода и накопления информации. С помощью этих ЭВМ создавались первые экспериментальные вычислительные системы и сети.

Компьютеры третьего поколения стали широко использоваться в качестве технической базы для самых различных автоматизированных систем - бухгалтерских и банковских систем, банков данных, систем автоматизации проектирования и производства и т. п. В это время появились первые администраторы баз данных и информационные службы по эксплуатации автоматизированных систем.

IV поколение, после 1975 года

Что характерно для машин четвёртого поколения?

Четвёртое поколение — это теперешнее поколение компьютерной техники, разработанное после 1970 года.

Наиболее важный в концептуальном отношении критерий, по которому эти компьютеры можно отделить от машин третьего поколения, состоит в том, что машины четвёртого поколения проектировались в расчете на эффективное использование современных высокоуровневых языков и упрощение процесса программирования для конечного пользователя.

В аппаратурном отношении для них характерно широкое использование *интегральных схем* в качестве элементной базы, а также наличие быстродействующих запоминающих устройств с произвольной выборкой ёмкостью в десятки мегабайт.

С точки зрения структуры машины этого поколения представляют собой *многопроцессорные* и *многомашинные комплексы*, работающие на общую память и общее поле внешних устройств. Быстродействие составляет до нескольких десятков миллионов операций в секунду, ёмкость оперативной памяти порядка 1 - 64 Мбайт.

Для них характерны:

- применение персональных компьютеров;
- телекоммуникационная обработка данных;
- компьютерные сети;
- широкое применение систем управления базами данных;
- элементы интеллектуального поведения систем обработки данных и устройств.

Использование при создании компьютеров больших интегральных схем (БИС - 1000 - 100000 компонентов на кристалл) и сверхбольших интегральных схем (СБИС - 100000 - 10000000 компонентов на кристалл).

Началом данного поколения считают 1975 год - фирма Amdahl Corp. выпустила шесть компьютеров AMDAHL 470 V/6, в которых были применены БИС в качестве элементной базы.

Стали использоваться быстродействующие системы памяти на интегральных схемах - МОП ЗУПВ емкостью в несколько мегабайт. В случае выключения машины данные, содержащиеся в МОП ЗУПВ, сохраняются путем автоматического переноса на диск. При включении машины запуск системы осуществляется при помощи хранимой в ПЗУ (постоянное запоминающее устройство) программы самозагрузки, обеспечивающей выгрузку операционной системы и резидентного программного обеспечения в МОП ЗУПВ. В середине 70-х появились первые персональные компьютеры.

Четвертое поколение - это компьютеры, создаваемые на базе серийных микропроцессоров. С этого поколения ЭВМ началось массовое производство и распространение персональных компьютеров, которые могут устанавливаться на любом рабочем столе - дома, на работе или в офисе.

Персональные ЭВМ широко используются для учебы, игры, написания писем, книг и отчетов, ведения бухгалтерской документации и экономических расчетов, проведения научных и маркетинговых исследований, сочинения стихов и музыки, ведения переписки с коллегами и друзьями.

Применение компьютеров в жизни общества затрагивает условия деятельности и жизни миллионов людей. Современные персональные компьютеры прежде всего открывают возможность выхода в сеть Интернет и оперативного поиска и получения различной информации в форме электронной почты, электронных журналов, газет и библиотек из самых различных стран и регионов, электронной коммерции - покупок и продаж по всему миру.

В серии ЭВМ четвертого поколения используются и более мощные компьютеры, получившие название **серверов** - вычислительных машин с большим объемом памяти, используемых для постоянного хранения больших объемов информации. Именно такие серверы и используются в качестве узлов связи в вычислительных системах и сети Интернет.

V поколение

Какими должны быть компьютеры пятого поколения?

Разработка последующих поколений компьютеров производится на основе больших интегральных схем повышенной степени интеграции, использования оптоэлектронных принципов (лазеры, голография).

Развитие идет также по пути *"интеллектуализации"* компьютеров, устранения барьера между человеком и компьютером. Компьютеры будут способны воспринимать информацию с рукописного или печатного текста, с бланков, с человеческого голоса, узнавать пользователя по голосу, осуществлять перевод с одного языка на другой.

В компьютерах пятого поколения произойдёт качественный переход от обработки *данных* к обработке *знаний*.

Архитектура компьютеров будущего поколения будет содержать два основных блока. Один из них — это традиционный компьютер. Но теперь он лишён связи с пользователем. Эту связь осуществляет блок, называемый термином "интеллектуальный интерфейс". Его задача — понять текст, написанный на естественном языке и содержащий условие задачи, и перевести его в работающую программу для компьютера.

Будет также решаться проблема децентрализации вычислений с помощью компьютерных сетей, как больших, находящихся на значительном расстоянии друг от друга, так и миниатюрных компьютеров, размещённых на одном кристалле полупроводника.

Главный упор при создании компьютеров сделан на их "интеллектуальность", внимание акцентируется не столько на элементной базе, сколько на переходе от архитектуры, ориентированной на обработку данных, к архитектуре, ориентированной на обработку знаний.

Обработка знаний - использование и обработка компьютером знаний, которыми владеет человек для решения проблем и принятия решений.

Академик В.М. Глушков еще в начале 80-х годов писал, что «к началу следующего столетия в развитых странах основная масса информации будет храниться в памяти ЭВМ, а человек XXI века, который не будет уметь пользоваться ЭВМ, будет подобен человеку XX века, не умевшему ни читать, ни писать».

Развитие вычислительной техники.

В ее развитии отмечают *предысторию* и *четыре поколения ЭВМ*. Предыстория начинается в глубокой древности с различных приспособлений для счета (абак, счеты), а первая счетная машина появилась лишь в 1642 г. Ее изобрел французский математик *Паскаль*. Построенная на основе зубчатых колес, она могла суммировать десятичные числа. Все четыре арифметические действия выполняла машина, созданная в 1673 г. немецким математиком *Лейбницем*. Она стала прототипом арифмометров, использовавшихся с 1820 г. до 60-х годов XX в. Впервые идея программно-управляемой счетной машины, имеющей арифметическое устройство, устройства управления, ввода и печати (хотя и использующей десятичную систему счисления), была выдвинута в 1822 г. английским математиком *Бэббиджем*. Его проект опережал технические возможности своего времени и не был реализован. Лишь в 40-х годах XX в. удалось создать программируемую счетную машину, причем на основе электромеханических реле, которые могут пребывать в одном из двух устойчивых состояний: "включено" и "выключено". Это технически проще, чем пытаться реализовать десять различных состояний, опирающихся на обработку информации на основе десятичной, а не двоичной системы счисления. Во второй половине 40-х годов появились первые электронно-вычислительные машины, элементной базой которых были электронные лампы. Основные характеристики ЭВМ разных поколений приведены в табл. 1.

С каждым новым поколением ЭВМ увеличивались быстродействие и надежность их работы при уменьшении стоимости и размеров, совершенствовались устройства ввода и вывода информации. В соответствии с трактовкой компьютера — как технической модели информационной функции человека — устройства ввода приближаются к естественному для человека восприятию информации (зрительному, звуковому) и, следовательно, операция по ее вводу в компьютер становится все более удобной для человека.

Современный компьютер — это универсальное, многофункциональное, электронное автоматическое устройство для работы с информацией. Компьютеры в современном обществе взяли на себя значительную часть работ, связанных с информацией. По историческим меркам компьютерные технологии обработки информации еще очень молоды и находятся в самом начале своего развития. Еще ни одно государство на Земле не создало информационного общества. Еще много потоков информации, не вовлеченных в сферу действия компьютеров. Компьютерные технологии сегодня преобразуют или вытесняют старые, докомпьютерные технологии обработки информации. Текущий этап завершится построением в индустриально развитых странах глобальных всемирных сетей для хранения и обмена информацией, доступных каждой организации и каждому члену общества. Надо только помнить, что компьютерам следует поручать то, что они могут делать лучше человека, и не употреблять во вред человеку, обществу.

Основные термины

Микросхема, транзистор, лампа, операционные системы, высокоуровневые языки, транслятор, первое поколение ЭВМ, второе поколение, машины третьего поколения, четвёртое поколение, пятое поколение, развитие вычислительной техники.

К о н т р о л ь н ы е в о п р о с ы

1. По каким признакам можно разделять компьютеры на классы и виды?
2. Как эволюционировала элементная база компьютеров от поколения к поколению?
3. В какой последовательности возникали известные Вам языки программирования?
4. Когда микрокомпьютеры стали доступны для широкого домашнего применения?
5. Можете ли Вы связать понятия "яблоко", "гараж" и "компьютер"?
6. На основе, каких технических элементов создавались компьютеры первого поколения?
7. Какую основную проблему перед разработчиками и пользователями выдвинул опыт эксплуатации компьютеров первого поколения?
8. Какая элементная база характерна для второго поколения компьютеров?
9. Какую функцию выполняет операционная система в процессе работы компьютера?
10. На какой элементной базе конструируются машины третьего поколения?
11. Из каких основных этапов состоит процесс изготовления микросхем?
12. Для каких поколений компьютеров характерно широкое использование интегральных схем?
13. Какое быстродействие характерно для машин четвёртого поколения?
14. Что подразумевают под "интеллектуальностью" компьютеров?
15. Какую задачу должен решать "интеллектуальный интерфейс" в машинах пятого поколения?

Тема 3. Арифметические основы компьютеров

План:

1. Что такое система счисления?
2. Двоичная система счисления.
3. Перевод чисел из десятичной системы в двоичную и наоборот
4. Восьмеричная система счисления.
5. Операции сложения и умножения в системах счислениях.

Что такое система счисления?

Система счисления — это способ записи чисел с помощью заданного набора специальных знаков (цифр).

Существуют позиционные и непозиционные системы счисления.

В *непозиционных* системах вес цифры (т.е. тот вклад, который она вносит в значение числа) не зависит от ее позиции в записи числа. Так, в римской системе счисления в числе XXXII (тридцать два) вес цифры X в любой позиции равен просто десяти.

В *позиционных* системах счисления вес каждой цифры изменяется в зависимости от ее положения (позиции) в последовательности цифр, изображающих число. Например, в числе 757,7 первая семерка означает 7 сотен, вторая – 7 единиц, а третья – 7 десятых долей единицы.

Сама же запись числа 757,7 означает сокращенную запись выражения

$700 + 50 + 7 + 0{,}7 = 7 \cdot 10^2 + 5 \cdot 10^1 + 7 \cdot 10^0 + 7 \cdot 10^{-1} = 757{,}7.$

Любая позиционная система счисления характеризуется своим основанием.

Основание позиционной системы счисления — это количество различных знаков или символов, используемых для изображения цифр в данной системе.

За основание системы можно принять любое натуральное число — два, три, четыре и т.д. Следовательно, возможно бесчисленное множество позиционных систем: двоичная, троичная, четверичная и т.д. Запись чисел в каждой из систем счисления с основанием q означает сокращенную запись выражения

$$a_{n-1} q^{n-1} + a_{n-2} q^{n-2} + \ldots + a_1 q^1 + a_0 q^0 + a_{-1} q^{-1} + \ldots + a_{-m} q^{-m},$$

где a_i – цифры системы счисления; n и m – число целых и дробных разрядов, соответственно.

Как порождаются целые числа в позиционных системах счисления?

В каждой системе счисления цифры упорядочены в соответствии с их значениями: 1 больше 0, 2 больше 1 и т.д.

Продвижением цифры называют замену её следующей по величине.

Продвинуть цифру 1 значит заменить её на 2, продвинуть цифру 2 значит заменить её на 3 и т.д. Продвижение старшей цифры (например, цифры 9 в десятичной системе) означает замену её на 0. В *двоичной* системе, использующей только две цифры – 0 и 1, продвижение 0 означает замену его на 1, а продвижение 1 – замену её на 0.

Целые числа в любой системе счисления порождаются с помощью Правила счета [44]:

Для образования целого числа, следующего за любым данным целым числом, нужно *продвинуть* самую правую цифру числа; если какая-либо цифра после продвижения стала нулем, то нужно продвинуть цифру, стоящую слева от неё.

Применяя это правило, запишем первые десять целых чисел

- в двоичной системе: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001;
- в троичной системе: 0, 1, 2, 10, 11, 12, 20, 21, 22, 100;
- в пятеричной системе: 0, 1, 2, 3, 4, 10, 11, 12, 13, 14;
- восьмеричной системе: 0, 1, 2, 3, 4, 5, 6, 7, 10, 11.

Какие системы счисления используют специалисты для общения с компьютером?

Кроме десятичной широко используются системы с основанием, являющимся *целой степенью числа 2, а именно*:

- двоичная (используются цифры 0, 1);
- восьмеричная (используются цифры 0, 1, ..., 7);
- шестнадцатеричная (для первых целых чисел от нуля до девяти используются цифры 0, 1, ..., 9, а для следующих чисел — от десяти до пятнадцати – в качестве цифр используются символы A, B, C, D, E, F).

Полезно запомнить запись в этих системах счисления первых двух десятков целых чисел:

10 - я	2 - я	8 - я	16 - я
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9

10 - я	2 - я	8 - я	16 - я
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13

Из всех систем счисления особенно проста и поэтому интересна для технической реализации в компьютерах двоичная система счисления.

Почему люди пользуются десятичной системой, а компьютеры — двоичной?

Люди предпочитают десятичную систему, вероятно, потому, что с древних времен считали по пальцам, а пальцев у людей по десять на руках и ногах. Не всегда и не везде люди пользуются десятичной системой счисления. В Китае, например, долгое время пользовались пятеричной системой счисления.

А компьютеры используют двоичную систему потому, что она имеет ряд преимуществ перед другими системами:

- для ее реализации нужны технические устройства с двумя устойчивыми состояниями (есть ток — нет тока, намагничен — не намагничен и т.п.), а не, например, с десятью, — как в десятичной;
- представление информации посредством только двух состояний надежно и помехоустойчиво;
- возможно применение аппарата булевой алгебры для выполнения логических преобразований информации;
- двоичная арифметика намного проще десятичной.

Недостаток двоичной системы — быстрый рост числа разрядов, необходимых для записи чисел.

Почему в компьютерах используются также восьмеричная и шестнадцатеричная системы счисления?

Двоичная система, удобная для компьютеров, для человека неудобна из-за ее *громоздкости* и *непривычной записи*.

Перевод чисел из десятичной системы в двоичную и наоборот выполняет машина. Однако, чтобы профессионально использовать компьютер, следует научиться понимать слово машины. Для этого и разработаны восьмеричная и шестнадцатеричная системы.

Числа в этих системах читаются почти так же легко, как десятичные, требуют соответственно *в три* (восьмеричная) и *в четыре* (шестнадцатеричная) *раза меньше разрядов, чем в двоичной системе* (ведь числа 8 и 16 – соответственно, третья и четвертая степени числа 2).

Перевод *восьмеричных* и *шестнадцатеричных* чисел *в двоичную систему* очень прост: достаточно каждую цифру заменить эквивалентной ей двоичной *триадой* (тройкой цифр) или *тетрадой* (четверкой цифр).

Например:

Чтобы перевести число из двоичной системы в восьмеричную или шестнадцатеричную, его нужно разбить влево и вправо от запятой на триады (для восьмеричной) или тетрады (для шестнадцатеричной) и каждую такую группу заменить соответ-ствующей восьмеричной (шестнадцатеричной) цифрой.

Как перевести целое число из десятичной системы в любую другую позиционную систему счисления?

При переводе целого *десятичного* числа в систему с основанием q его необходимо последовательно *делить* на q до тех пор, пока не останется остаток, меньший или равный $q–1$. Число в системе с основанием q записывается как последо-вательность остатков от деления, записанных в обратном порядке, начиная с последнего.

Пример: Перевести число 75 из десятичной системы в двоичную, восьмеричную и шестнадцатеричную:

Ответ: $75_{10} = 1\ 001\ 011_2 = 113_8 = 4B_{16}$.

Как перевести правильную десятичную дробь в любую другую позиционную систему счисления?

При переводе *правильной десятичной дроби* в систему счисления с основанием q необходимо сначала саму дробь, а затем дробные части всех последующих произведений последовательно *умножать* на q, отделяя после каждого умножения целую часть произведения. Число в новой системе счисления записывается как последовательность полученных целых частей произведения.

Умножение производится до тех пор, пока дробная часть произведения не станет равной нулю. Это значит, что сделан точный перевод. В противном случае перевод осуществляется до заданной точности. Достаточно того количества цифр в результате, которое поместится в ячейку.

Пример: Перевести число 0,35 из десятичной системы в двоичную, восьмеричную и шестнадцатеричную:

Ответ: $0{,}35_{10} = 0{,}01011_2 = 0{,}263_8 = 0{,}59_{16}$.

Как перевести число из двоичной (восьмеричной, шестнадцатеричной) системы в десятичную?

При переводе числа из двоичной (восьмеричной, шестнадцатеричной) системы в десятичную надо это число представить в виде суммы степеней основания его системы счисления.

Как производятся арифметические операции в позиционных системах счисления?

Рассмотрим основные арифметические операции: *сложение, вычитание, умножение* и *деление*. Правила выполнения этих операций в десятичной системе хорошо известны — это *сложение, вычитание, умножение столбиком* и *деление углом*. Эти правила применимы и ко всем другим позиционным системам счисления. Только *таблицами сложения и умножения надо пользоваться особыми для каждой системы*.

Сложение

Таблицы сложения легко составить, используя Правило Счета.

Сложение в двоичной системе	Сложение в восьмеричной системе

Сложение в шестнадцатиричной системе

При сложении цифры суммируются по разрядам, и если при этом возникает избыток, то он переносится влево.

Пример 1. Сложим числа 15 и 6 в различных системах счисления.

Шестнадцатеричная: $F_{16}+6_{16}$	**Ответ**: $15+6 = 21_{10} = 10101_2 = 25_8 = 15_{16}$. **Проверка. Преобразуем полученные суммы к десятичному виду:** $10101_2 = 2^4 + 2^2 + 2^0 = 16+4+1=21$, $25_8 = 2*8^1 + 5*8^0 = 16 + 5 = 21$, $15_{16} = 1*16_1 + 5*16_0 = 16+5 = 21$.

Пример 2. Сложим числа 15, 7 и 3.

Шестнадцатеричная: $F_{16}+7_{16}+3_{16}$	**Ответ**: $5+7+3 = 25_{10} = 11001_2 = 31_8 = 19_{16}$. **Проверка:** $11001_2 = 2^4 + 2^3 + 2^0 = 16+8+1=25$, $31_8 = 3*8^1 + 1*8^0 = 24 + 1 = 25$, $19_{16} = 1*16^1 + 9*16^0 = 16+9 = 25$.

Пример 3. Сложим числа 141,5 и 59,75.

Ответ: $141,5 + 59,75 = 201,25_{10} = 11001001,01_2 = 311,2_8 = C9,4_{16}$

Проверка. Преобразуем полученные суммы к десятичному виду:

$11001001,01_2 = 2^7 + 2^6 + 2^3 + 2^0 + 2^{-2} = 201,25$

$311,2_8 = 3*8^2 + 1•8^1 + 1*8^0 + 2*8^{-1} = 201,25$

$C9,4_{16} = 12*16^1 + 9*16^0 + 4*16^{-1} = 201,25$

Вычитание

Пример 4. Вычтем единицу из чисел 10_2, 10_8 и 10_{16}

Пример 5. Вычтем единицу из чисел 100_2, 100_8 и 100_{16}.

Пример 6. Вычтем число 59,75 из числа 201,25.

Ответ: $201,25_{10} – 59,75_{10} = 141,5_{10} = 10001101,1_2 = 215,4_8 = 8D,8_{16}$.

Проверка. Преобразуем полученные разности к десятичному виду:

$10001101,1_2 = 2^7 + 2^3 + 2^2 + 2^0 + 2^{-1} = 141,5$;

$215,4_8 = 2*8^2 + 1*8^1 + 5*8^0 + 4*8^{-1} = 141,5$;

$8D,8_{16} = 8*16^1 + D*16^0 + 8*16^{-1} = 141,5$.

Умножение

Выполняя умножение многозначных чисел в различных позиционных системах счисления, можно использовать обычный алгоритм перемножения чисел в столбик, но при этом результаты перемножения и сложения однозначных чисел необходимо заимствовать из соответствующих рассматриваемой системе таблиц умножения и сложения.

Умножение в двоичной системе	Умножение в восьмеричной системе

Ввиду чрезвычайной простоты таблицы умножения в двоичной системе, умножение сводится лишь к сдвигам множимого и сложениям.

Пример 7. Перемножим числа 5 и 6.

Ответ: $5*6 = 30_{10} = 11110_2 = 36_8$.

Проверка. Преобразуем полученные произведения к десятичному виду:

$11110_2 = 2^4 + 2^3 + 2^2 + 2^1 = 30$;

$36_8 = 3•8^1 + 6•8^0 = 30$.

Пример 8. Перемножим числа 115 и 51.

Ответ: $115*51 = 5865_{10} = 1011011101001_2 = 13351_8$.

Проверка. Преобразуем полученные произведения к десятичному виду:

$1011011101001_2 = 2^{12} + 2^{10} + 2^9 + 2^7 + 2^6 + 2^5 + 2^3 + 2^0 = 5865$;

$13351_8 = 1*8^4 + 3*8^3 + 3*8^2 + 5*8^1 + 1*8^0 = 5865$.

Деление

Деление в любой позиционной системе счисления производится по тем же правилам, как и деление углом в десятичной системе. В двоичной системе деление выполняется особенно просто, ведь очередная цифра частного может быть только нулем или единицей.

Пример 9. Разделим число 30 на число 6.

Ответ: $30: 6 = 5_{10} = 101_2 = 5_8$.

Пример 10. Разделим число 5865 на число 115.

Восьмеричная: $13351_8 :163_8$

Ответ: $5865: 115 = 51_{10} = 110011_2 = 63_8$.

Проверка. Преобразуем полученные частные к десятичному виду:

$110011_2 = 2^5 + 2^4 + 2^1 + 2^0 = 51$; $63_8 = 6*8^1 + 3*8^0 = 51$.

Пример 11. Разделим число 35 на число 14.

Восьмеричная: $43_8 : 16_8$

Ответ: $35 : 14 = 2{,}5_{10} = 10{,}1_2 = 2{,}4_8$.

Проверка. Преобразуем полученные частные к десятичному виду:

$10{,}1_2 = 2^1 + 2^{-1} = 2{,}5$;

$2{,}4_8 = 2*8^0 + 4*8^{-1} = 2{,}5$.

Как представляются в компьютере целые числа?

Целые числа могут представляться в компьютере со знаком или без знака.

Целые числа без знака обычно занимают в памяти один или два байта и принимают в однобайтовом формате значения от 00000000_2 до 11111111_2 , а в двухбайтовом формате — от 00000000 00000000_2 до 11111111 11111111_2.

Диапазоны значений целых чисел без знака

Формат числа в байтах	Диапазон	
	Запись с порядком	Обычная запись
1	$0 \ldots 2^8–1$	0 … 255
2	$0 \ldots 2^{16}–1$	0 … 65535

Примеры:

а) число $72_{10} = 1001000_2$ в однобайтовом формате:

б) это же число в двухбайтовом формате:

в) число 65535 в двухбайтовом формате:

Целые числа со знаком обычно занимают в памяти компьютера один, два или четыре байта, при этом самый левый (старший) разряд содержит информацию о знаке числа. Знак “плюс” кодируется нулем, а “минус” — единицей.

Диапазоны значений целых чисел со знаком

Формат числа в байтах	Диапазон	
	Запись с порядком	Обычная запись
1	$-2^7 \dots 2^7-1$	–128 … 127
2	$-2^{15} \dots 2^{15}-1$	–32768 … 32767
4	$-2^{31} \dots 2^{31}-1$	–2147483648 … 2147483647

Рассмотрим особенности записи целых чисел со знаком на примере однобайтового формата, при котором для знака отводится один разряд, а для цифр абсолютной величины – семь разрядов.

В компьютерной технике применяются три формы записи (кодирования) целых чисел со знаком: *прямой* код, *обратный* код, *дополнительный* код.

Последние две формы применяются особенно широко, так как позволяют упростить конструкцию арифметико-логического устройства компьютера путем замены разнообразных арифметических операций операцией сложения.

Положительные числа в прямом, обратном и дополнительном кодах изображаются одинаково — двоичными кодами с цифрой 0 в знаковом разряде.

Отрицательные числа в прямом, обратном и дополнительном кодах имеют разное изображение.

1. Прямой код. В знаковый разряд помещается цифра 1, а в разряды цифровой части числа — двоичный код его абсолютной величины.
2. Обратный код. Получается инвертированием всех цифр двоичного кода абсолютной величины числа, включая разряд знака: нули заменяются единицами, а единицы — нулями.
3. Дополнительный код. Получается образованием обратного кода с последующим прибавлением единицы к его младшему разряду.

Обычно *отрицательные* десятичные числа при вводе в машину *автоматически* преобразуются в *обратный* или *дополнительный* двоичный код и в таком виде хранятся, перемещаются и участвуют в операциях. При выводе таких чисел из машины происходит *обратное преобразование* в отрицательные десятичные числа.

Как компьютер выполняет арифметические действия над целыми числами?

Сложение и вычитание

В большинстве компьютеров *операция вычитания не используется*. Вместо нее производится *сложение* уменьшаемого с *обратным* или *дополнительным* кодом вычитаемого. Это позволяет существенно упростить конструкцию АЛУ.

При сложении обратных кодов чисел А и В имеют место четыре основных и два особых случая:

1. А и В положительные. При суммировании складываются все разряды, включая разряд знака. Так как знаковые разряды положительных слагаемых равны нулю, разряд знака суммы тоже равен нулю.

Например:

Получен правильный результат.

2. А положительное, В отрицательное и по абсолютной величине больше, чем А. Например:

Получен правильный результат в обратном коде. При переводе в прямой код биты цифровой части результата инвертируются: 1 0000111 = -7_{10}.

3. А положительное, В отрицательное и по абсолютной величине меньше, чем А. Например:

Компьютер исправляет полученный первоначально неправильный результат (6 вместо 7) переносом единицы из знакового разряда в младший разряд суммы.

4. А и В отрицательные.

Например:

Полученный первоначально неправильный результат (обратный код числа -11_{10} вместо обратного кода числа -10_{10}) компьютер исправляет переносом единицы из знакового разряда в младший разряд суммы. При переводе результата в прямой код биты цифровой части числа инвертируются: 1 0001010 = -10_{10}. При сложении может возникнуть ситуация, когда старшие разряды результата операции не помещаются в отведенной для него области памяти. Такая ситуация называется *переполнением разрядной сетки формата числа*. Для обнаружения переполнения и

оповещения о возникшей ошибке в компьютере используются специальные средства. Ниже приведены два возможных случая переполнения.

5. A и B положительные, сумма A+B больше, либо равна 2^{n-1}, где n – количество разрядов формата чисел (для однобайтового формата n=8, $2^{n-1} = 27 = 128$).

Например:

Семи разрядов цифровой части числового формата недостаточно для размещения восьмиразрядной суммы ($162_{10} = 10100010_2$), поэтому старший разряд суммы оказывается в знаковом разряде. Это вызывает несовпадение знака суммы и знаков слагаемых, что является свидетельством переполнения разрядной сетки.

6. A и B отрицательные, сумма абсолютных величин A и B больше, либо равна 2^{n-1}. Например:

Здесь знак суммы тоже не совпадает со знаками слагаемых, что свидетельствует о переполнении разрядной сетки.

Все эти случаи имеют место и при сложении *дополнительных кодов* чисел:

1. A и B положительные. Здесь нет отличий от случая 1, рассмотренного для обратного кода.
2. A положительное, B отрицательное и по абсолютной величине больше, чем A. Например:
6. Получен правильный результат в дополнительном коде. При переводе в прямой код биты цифровой части результата инвертируются и к младшему разряду прибавляется единица: 1 0000110 + 1 = 1 0000111 = -7_{10}.
3. A положительное, B отрицательное и по абсолютной величине меньше, чем A. Например:
7. Получен правильный результат. Единицу переноса из знакового разряда компьютер отбрасывает.
4. A и B отрицательные. Например:
8. Получен правильный результат в дополнительном коде. Единицу переноса из знакового разряда компьютер отбрасывает.

Случаи переполнения для дополнительных кодов рассматриваются по аналогии со случаями 5 и 6 для обратных кодов.

Сравнение рассмотренных форм кодирования целых чисел со знаком показывает:

- на преобразование отрицательного числа в обратный код компьютер затрачивает меньше времени, чем на преобразование в дополнительный код, так как последнее состоит из двух шагов — образования обратного кода и прибавления единицы к его младшему разряду;
- время выполнения сложения для дополнительных кодов чисел меньше, чем для их обратных кодов, потому что в таком сложении нет переноса единицы из знакового разряда в младший разряд результата.

Умножение и деление

Во многих компьютерах умножение производится как последовательность сложений и сдвигов. Для этого в АЛУ имеется регистр, называемый накапливающим сумматором, который до начала выполнения операции содержит число ноль. В процессе выполнения операции в нем поочередно размещаются множимое и результаты промежуточных сложений, а по завершении операции — окончательный результат.

Другой регистр АЛУ, участвующий в выполнении этой операции, вначале содержит множитель. Затем по мере выполнения сложений содержащееся в нем число уменьшается, пока не достигнет нулевого значения.

Для иллюстрации умножим 110011_2 на 101101_2.

Деление для компьютера является трудной операцией. Обычно оно реализуется путем многократного прибавления к делимому дополнительного кода делителя.

Как представляются в компьютере вещественные числа?

Вещественными числами (в отличие от целых) в компьютерной технике называются числа, имеющие дробную часть.

При их написании вместо запятой принято писать точку. Так, например, число 5 — целое, а числа 5.1 и 5.0 — вещественные.

Для удобства отображения чисел, принимающих значения из достаточно широкого диапазона (то есть, как очень маленьких, так и очень больших), используется форма записи чисел с порядком основания системы счисления. Например, десятичное число 1.25 можно в этой форме представить так:

$$1.25*10^0 = 0.125*10^1 = 0.0125*10^2 = \dots ,$$

или так:

$$12.5*10^{-1} = 125.0*10^{-2} = 1250.0*10^{-3} = \dots .$$

Любое число N в системе счисления с основанием q можно записать в виде $N = M * q^p$, где M называется мантиссой числа, а p — порядком. Такой способ записи чисел называется представлением с плавающей точкой.

Если “плавающая” точка расположена в мантиссе перед первой значащей цифрой, то при фиксированном количестве разрядов, отведённых под мантиссу, обеспечивается запись максимального количества значащих цифр числа, то есть максимальная точность представления числа в машине. Из этого следует:

Мантисса должна быть правильной дробью, первая цифра которой отлична от нуля: M из [0.1, 1).

Такое, наиболее выгодное для компьютера, представление вещественных чисел называется нормализованным.

Мантиссу и порядок q-ичного числа принято записывать в системе с основанием q, а самооснование — в десятичной системе.

Примеры нормализованного представления:

Десятичная система	Двоичная система
$753.15 = 0.75315*10^3$;	$-101.01 = -0.10101*2^{11}$ (порядок $11_2 = 3_{10}$)
$-0.000034 = -0.34*10^{-4}$;	$-0.000011 = 0.11*2^{-100}$ (порядок $-100_2 = -410$)

Вещественные числа в компьютерах различных типов записываются по-разному. При этом компьютер обычно предоставляет программисту возможность выбора из нескольких числовых форматов наиболее подходящего для конкретной задачи — с использованием четырех, шести, восьми или десяти байтов.

В качестве примера приведем характеристики форматов вещественных чисел, используемых IBM-совместимыми персональными компьютерами:

Форматы вещественных чисел	Размер в байтах	Примерный диапазон абсолютных значений	Количество значащих десятичных цифр
Одинарный	4	$10^{-45} \dots 10^{38}$	7 или 8
Вещественный	6	$10^{-39} \dots 10^{38}$	11 или 12
Двойной	8	$10^{-324} \dots 10^{308}$	15 или 16
Расширенный	10	$10^{-4932} \dots 10^{4932}$	19 или 20

Из этой таблицы видно, что форма представления чисел с плавающей точкой позволяет записывать числа с высокой точностью и из весьма широкого диапазона.

При хранении числа с плавающей точкой отводятся разряды для мантиссы, порядка, знака числа и знака порядка:

- Чем больше разрядов отводится под запись мантиссы, тем выше точность представления числа.
- Чем больше разрядов занимает порядок, тем шире диапазон от наименьшего отличного от нуля числа до наибольшего числа, представимого в машине при заданном формате.

Покажем на примерах, как записываются некоторые числа в нормализованном виде в четырехбайтовом формате с семью разрядами для записи порядка.

1. Число $6.25_{10} = 110.01_2 = 0{,}11001 \cdot 2^{11}$:
2. Число $-0.125_{10} = -0.0012 = -0.1*2^{-10}$

(отрицательный порядок записан в дополнительном коде):

Как компьютер выполняет арифметические действия над нормализованными числами?

К началу выполнения арифметического действия операнды операции помещаются в соответствующие регистры АЛУ.

Сложение и вычитание

При сложении и вычитании сначала производится подготовительная операция, называемая выравниванием порядков.

В процессе выравнивания порядков мантисса числа с меньшим порядком *сдвигается в своем регистре вправо* на количество разрядов, равное разности порядков операндов. После каждого сдвига *порядок увеличивается на единицу*.

В результате выравнивания порядков одноименные разряды чисел оказываются расположенными в соответствующих разрядах обоих регистров, после чего мантиссы складываются или вычитаются.

В случае необходимости полученный результат нормализуется путем сдвига мантиссы результата влево. После каждого сдвига влево порядок результата уменьшается на единицу.

Пример 1. Сложить двоичные нормализованные числа $0.10111 \cdot 2^{-1}$ и $0.11011 * 2^{10}$. Разность порядков слагаемых здесь равна трем, поэтому перед сложением мантисса первого числа сдвигается на три разряда вправо:

Пример 2. Выполнить вычитание двоичных нормализованных чисел $0.10101 * 2^{10}$ и $0.11101 * 2^{1}$. Разность порядков уменьшаемого и вычитаемого здесь равна единице, поэтому перед вычитанием мантисса второго числа сдвигается на один разряд вправо:

Результат получился не нормализованным, поэтому его мантисса сдвигается влево на два разряда с соответствующим уменьшением порядка на две единицы: $0.1101 * 2^{0}$.

Умножение

При умножении двух нормализованных чисел их порядки складываются, а мантиссы перемножаются.

Пример 3. Выполнить умножение двоичных нормализованных чисел:
$(0.11101 * 2^{101}) * (0.1001 * 2^{11}) = (0.11101 * 0.1001) * 2^{(101+11)} = 0.100000101 * 2^{1000}$.

Деление

При делении двух нормализованных чисел из порядка делимого вычитается порядок делителя, а мантисса делимого делится на мантиссу делителя. Затем в случае необходимости полученный результат нормализуется.

Пример 4. Выполнить деление двоичных нормализованных чисел:
$0.1111 * 2^{100} : 0.101 * 2^{11} = (0.1111 : 0.101) * 2^{(100-11)} = 1.1 * 2^{1} = 0.11 \cdot 2^{10}$.

Использование представления чисел с плавающей точкой существенно усложняет схему арифметико-логического устройства.

Основные термины

Система счисления, целые числа, десятичная система, двоичная система, восьмеричная и шестнадцатеричная система, позиционные и непозиционные системы счисления, арифметические операции, вещественные числа, нормализованные числа.

К о н т р о л ь н ы е в о п р о с ы

1. Что такое система счисления?
2. Как порождаются целые числа в позиционных системах счисления?
3. Какие системы счисления используют специалисты для общения с компьютером?
4. Почему люди пользуются десятичной системой, а компьютеры — двоичной?
5. Почему в компьютерах используются также восьмеричная и шестнадцатеричная системы счисления?
6. Как перевести целое число из десятичной системы в любую другую позиционную систему счисления?
7. Как перевести правильную десятичную дробь в любую другую позиционную систему счисления?
8. Как перевести число из двоичной (восьмеричной, шестнадцатеричной) системы в десятичную?
9. Как производятся арифметические операции в позиционных системах счисления?
10. Как представляются в компьютере целые числа?
11. Как компьютер выполняет арифметические действия над целыми числами?
12. Как представляются в компьютере вещественные числа?

13. Как компьютер выполняет арифметические действия над нормализованными числами?

Тема 4. Представление информации в ЭВМ.

Персональные компьютеры.

План:

1. Единицы измерения информации.
2. Как измеряется количество информации?
3. Персональные компьютеры

Память компьютера построена из двоичных запоминающих элементов — *битов*, объединенных в группы по 8 битов, которые называются *байтами*. (Единицы измерения памяти совпадают с единицами измерения информации). Все байты пронумерованы. Номер байта называется его *адресом*.

Единица измерения информации называется бит (bit) – сокращение от английских слов binary digit, что означает двоичная цифра.

Минимальной единицей информации считается бит. **Бит** - это величина, принимающая значение 0 или 1. Любая другая информация может быть закодирована последовательностью из нулей и единиц. Именно в таком виде вся информация представляется в памяти ЭВМ.

Единицей памяти в современных ЭВМ считается байт. **Байты** - это 8-разрядные двоичные числа вида - 00000000, 00000001, ..., 11111111. Один байт записывается в виде 8 двоичных знаков информации - нулей и единиц:

1 байт = 8 бит.

Скорость передачи информации по линиям связи оценивается в бодах и килободах. Скорость в один бод - это передача одного бита в секунду:

1 бод = 1 бит/секунда.

1 Кбод = 1024 бод.

Байты могут объединяться в ячейки, которые называются также *словами*. Для каждого компьютера характерна определенная длина слова — два, четыре или восемь байтов. Это не исключает использования ячеек памяти другой длины (например, полуслово, двойное слово).

Как правило, в одном машинном слове может быть представлено либо одно целое число, либо одна команда. Однако допускаются переменные форматы представления информации.

Разбиение памяти на слова для четырехбайтовых компьютеров представлено в таблице:

Байт 0	Байт 1	Байт 2	Байт 3	Байт 4	Байт 5	Байт 6	Байт 7
ПОЛУСЛОВО		ПОЛУСЛОВО		ПОЛУСЛОВО		ПОЛУСЛОВО	
СЛОВО				СЛОВО			
ДВОЙНОЕ СЛОВО							

Широко используются и более крупные производные единицы объема памяти: *Килобайт*, *Мегабайт*, *Гигабайт*, а также, в последнее время, *Терабайт* и *Петабайт*.

Как измеряется количество информации?

Какое количество информации содержится, к примеру, в тексте романа "Война и мир", в фресках Рафаэля или в генетическом коде человека? Ответа на эти вопросы наука не даёт и, по всей вероятности, даст не скоро.

А возможно ли объективно измерить количество информации? Важнейшим результатом теории информации является вывод:

В определенных, весьма широких условиях можно пренебречь качественными особенностями информации, выразить её количество числом, а также сравнить количество информации, содержащейся в различных группах данных.

В настоящее время получили распространение подходы к определению понятия "количество информации", основанные на том, что информацию, содержащуюся в сообщении, можно нестрого трактовать в смысле её новизны или, иначе, уменьшения неопределённости наших знаний об объекте.

Так, американский инженер Р. Хартли (1928 г.) процесс получения информации рассматривает как выбор одного сообщения из конечного наперёд заданного множества из N равновероятных сообщений, а количество информации I, содержащееся в выбранном сообщении, определяет как двоичный логарифм N.

Формула Хартли: $I = \log_2 N$

Допустим, нужно угадать одно число из набора чисел от единицы до ста. По формуле Хартли можно вычислить, какое количество информации для этого требуется: $I = \log_2 100 \approx 6{,}644$. То есть сообщение о верно угаданном числе содержит количество информации, приблизительно равное 6,644 единиц информации.

Приведем другие примеры равновероятных сообщений:

1. при бросании монеты: *"выпала решка"*, *"выпал орел"*;
2. на странице книги: *"количество букв чётное"*, *"количество букв нечётное"*.

Определим теперь, являются ли равновероятными сообщения *"первой выйдет из дверей здания женщина"* и *"первым выйдет из дверей здания мужчина"*. Однозначно ответить на этот вопрос нельзя. Все зависит от того, о каком именно здании идет речь. Если это, например, станция метро, то вероятность выйти из дверей первым одинакова для мужчины и женщины, а если это военная казарма, то для мужчины эта вероятность значительно выше, чем для женщины.

Для задач такого рода американский учёный Клод Шеннон предложил в 1948 г. другую формулу определения количества информации, учитывающую возможную неодинаковую вероятность сообщений в наборе.

Формула Шеннона: $I = -(p_1 \log_2 p_1 + p_2 \log_2 p_2 + \ldots + p_N \log_2 p_N)$, где p_i — вероятность того, что именно *i*-е сообщение выделено в наборе из N сообщений.

Легко заметить, что если вероятности $p_1, \ldots, p_N$ равны, то каждая из них равна $1/N$, и формула Шеннона превращается в формулу Хартли.

Помимо двух рассмотренных подходов к определению количества информации, существуют и другие. Важно помнить, что любые теоретические результаты применимы лишь к определённому кругу случаев, очерченному первоначальными допущениями.

В качестве единицы информации условились принять один бит (*англ.* bit — binary, digit — двоичная цифра).

Бит в теории информации — количество информации, необходимое для различения двух равновероятных сообщений. А в вычислительной технике битом называют наименьшую "порцию" памяти, необходимую для хранения одного из двух знаков "0" и "1", используемых для внутримашинного представления данных и команд.

Бит — слишком мелкая единица измерения. На практике чаще применяется более крупная единица — *байт*, равная восьми битам. Именно восемь битов требуется для того, чтобы закодировать любой из 256 символов алфавита клавиатуры компьютера ($256=2^8$).

Широко используются также ещё более крупные производные единицы информации:

- 1 Килобайт (Кбайт) = 1024 байт = 2^{10} байт,
- 1 Мегабайт (Мбайт) = 1024 Кбайт = 2^{20} байт,
- 1 Гигабайт (Гбайт) = 1024 Мбайт = 2^{30} байт.

В последнее время в связи с увеличением объёмов обрабатываемой информации входят в употребление такие производные единицы, как:

- 1 Терабайт (Тбайт) = 1024 Гбайт = 2^{40} байт,
- 1 Петабайт (Пбайт) = 1024 Тбайт = 2^{50} байт.

За единицу информации можно было бы выбрать количество информации, необходимое для различения, например, десяти равновероятных сообщений. Это будет не двоичная (бит), а десятичная (дит) единица информации.

Персональные компьютеры

Компьютеры - это универсальные электронные вычислительные машины (ЭВМ), используемые для накопления, обработки и передачи информации. Самое широкое распространение получили персональные компьютеры, предназначенные для индивидуальной работы.

Персональные компьютеры - это малогабаритные вычислительные машины, которые могут быть установлены на любом рабочем месте. Наиболее известны и распространены персональные компьютеры **IBM PC** и **Macintosh.**

Минимальный состав персональных компьютеров:

1) системный блок;
2) дисплей;
3) клавиатура.

Дисплей - это устройство отображения информации на электронном экране. Дисплеи в персональных компьютерах могут быть цветными и черно-белыми. Информация на дисплеях обычно отображается как в телевизоре - на экране электронно-лучевой трубки.

Вид персонального компьютера

Клавиатура содержит клавиши, как правило, латинского и русского алфавитов. Кроме того, на клавиатуре имеются цифры и другие специальные знаки. Нажимая на эти клавиши, можно вводить в компьютер самую разную информацию - числа, слова, разы, а также команды управления компьютером.

Мышка - устройство, которое подсоединяется к персональному компьютеру электрическим шнуром и которое можно перемещать по столу.

Системный блок содержит процессор и оперативную память. Возможности компьютеров зависят от типа и быстродействия процессора, а также от объемов оперативной и долговременной памяти. Во всех современных персональных компьютерах в системный блок входят также накопители на магнитных дисках.

Процессор - это устройство управления компьютером. Быстродействие компьютеров определяется числом операций, выполняемых процессором за одну секунду. Основной функцией процессоров является автоматическое управление работой ЭВМ с помощью программ, размещаемых в оперативной памяти.

В компьютерах первого поколения быстродействие процессоров составляло несколько тысяч операций в секунду; второго поколения - несколько десятков тысяч, а в машинах третьего поколения - несколько сотен тысяч операций в секунду.

Быстродействие персональных компьютеров четвертого поколения - несколько миллионов операций в секунду. В компьютерах следующих поколений быстродействие будет составлять десятки и даже сотни миллионов операций в секунду.

В персональных компьютерах IBM PC используются процессоры фирмы Intel. В компьютерах младших моделей процессоры **Intel** - 86, 286, 386 и 486, а в старших моделях процессоры серии **Pentium** - Pentium, Pentium II, Pentium III, Pentium IV и т. д. В персональных компьютерах Macintosh применяются процессоры фирмы **Motorola.**

Программа - это последовательность команд и данных, которые могут интерпретироваться ЭВМ. Программы определяют конкретные функции и роли ЭВМ от игрового автомата и редактора текстов до рабочего места президента крупной фирмы или страны.

Машины первого поколения имели оперативную память порядка нескольких килобайт, компьютеры второго поколения - десятки килобайт, а машины третьего поколения - сотни килобайт.

Оперативная память в персональных компьютерах типа IBM PC и Macintosh составляет несколько мегабайт. В больших современных ЭВМ объем оперативной памяти достигает порядка десятков мегабайт, а в компьютерах новых поколений - сотни и тысячи мегабайт.

Для долговременного хранения информации и программ в персональных компьютерах используются **магнитные диски** - гибкие и жесткие. Информация в оперативной памяти удаляется после выключения компьютера. Информация на магнитных дисках может храниться после выключения ЭВМ до следующих сеансов работы.

Жесткие диски - это устройства хранения информации, программ и данных в ЭВМ. В персональных компьютерах жесткие диски находятся внутри системного блока и служат для постоянного хранения программ, данных, архивов и т.п.

Объем памяти на жестких дисках в современных компьютерах имеет диапазон от нескольких мегабайт до нескольких гигабайт. В компьютерах новых поколений объем памяти на жестких магнитных дисках будет составлять десятки и сотни гигабайт.

Гибкие диски - это сменные носители информации, на которых программы и данные можно хранить отдельно от ЭВМ. Гибкие диски используются для личного хранения и переноса программ и данных от одного компьютера к другому. Объем памяти на наиболее широко распространенных гибких магнитных дисках составляет от 360 Кбайт до 1,4 Мбайт.

К современным персональным компьютерам может быть подсоединен целый **ряд дополнительных устройств.** Наиболее часто к ним подключаются принтеры, модемы и компакт-дисководы. Компакт-дисковод - это устройство для считывания компакт-дисков.

Компакт-диск - это оптические диски с голографической записью информации. Особенность компакт-дисков - большой объем записанной на них информации, равной объему порядка 500 гибких дисков.

Компакт-диски - это средство для постоянного хранения информации, которая записывается один раз и может многократно считываться на ЭВМ. Компакт-диски - наиболее удобное средство для переноса больших объемов информации. Объем памяти на компакт-дисках составляет до 780 Мбайт.

Принтер - это печатающее устройство, подсоединяемое к компьютерам. Наибольшее распространение получили три типа принтеров, различающихся скоростью и качеством печати: матричные, струйные и лазерные. Самые простые и дешевые среди них - матричные, самые быстрые и качественные - лазерные, а струйные - самые качественные среди дешевых принтеров.

Модем - это устройство передачи информации по линиям телефонной связи. С помощью модемов персональные компьютеры могут подключаться через телефонную сеть к другим компьютерам, а также входить в различные телекоммуникационные компьютерные сети.

Основные термины

Единица измерения памяти, измерение количество информации, скорость передачи, персональные компьютеры, монитор, клавиатура, принтер, память, системный блок, мышка, процессор, модем, компакт диски.

К о н т р о л ь н ы е в о п р о с ы

1. Какие устройства входят в состав персональных компьютеров?
2. Что такое процессор?
3. Каково быстродействие современных процессоров?
4. В каких единицах измеряется объем памяти компьютеров?
5. Каков объем оперативной памяти современных компьютеров?
6. Каковы объемы памяти на гибких дисках?
7. Каковы объемы памяти на жестких дисках?
8. Каковы объемы памяти на компакт-дисках?

Тема 5. Общие принципы организации и работы компьютеров

План:

1. Компьютеры.
2. Как устроен компьютер?
3. На каких принципах построены компьютеры?
4. Команда и выполнение команда.
5. Архитектура и структура компьютера.
6. Центральный процессор.

Что такое компьютер?

Компьютер (англ. *computer* — вычислитель) представляет собой программируемое электронное устройство, способное обрабатывать данные и производить вычисления, а также выполнять другие задачи манипулирования символами [51].

Существует два основных класса компьютеров:

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

Поскольку в настоящее время подавляющее большинство компьютеров являются цифровыми, далее будем рассматривать только этот класс компьютеров и слово "*компьютер*" употреблять в значении "*цифровой компьютер*".

Основу компьютеров образует *аппаратура* (*HardWare*), построенная, в основном, с использованием электронных и электромеханических элементов и устройств. Принцип действия компьютеров состоит в выполнении *программ* (*SoftWare*) — заранее заданных, четко определённых последовательностей арифметических, логических и других операций.

Любая компьютерная программа представляет собой последовательность отдельных команд.

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

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

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

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

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

Как устроен компьютер?

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

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

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

Функции памяти:

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

Функции процессора:

- *обработка данных по заданной программе* путем выполнения арифметических и логических операций;
- *программное управление работой устройств* компьютера.

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

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

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

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

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

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

Существует несколько типов регистров, отличающихся видом выполняемых операций.

Некоторые важные регистры имеют свои названия, например:

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

На каких принципах построены компьютеры?

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

1. Принцип программного управления. Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.
2. Принцип однородности памяти. Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке памяти — число, текст или команда. Над командами можно выполнять такие же действия, как и над данными.
3. Принцип адресности. Структурно основная память состоит из перенумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка.

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

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

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

Что такое команда?

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

В общем случае, **команда содержит следующую информацию:**

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

В зависимости от количества операндов, **команды бывают:**

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

Команды хранятся в ячейках памяти *в двоичном коде*.

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

В адресной части команды может быть указан, например:

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

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

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

add	*x*

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

add	*x*	*y*

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

add	*x*	*y*	*z*

Как выполняется команда?

Как правило, этот процесс разбивается на следующие этапы:

- из ячейки памяти, адрес которой хранится в счетчике команд, выбирается очередная команда; содержимое счетчика команд при этом увеличивается на длину команды;
- выбранная команда передается в устройство управления на регистр команд;
- устройство управления расшифровывает адресное поле команды;
- по сигналам УУ операнды считываются из памяти и записываются в АЛУ на специальные регистры операндов;
- УУ расшифровывает код операции и выдает в АЛУ сигнал выполнить соответствующую операцию над данными;
- результат операции либо остается в процессоре, либо отправляется в память, если в команде был указан адрес результата;
- все предыдущие этапы повторяются до достижения команды "*стоп*".

Что такое архитектура и структура компьютера?

При рассмотрении компьютерных устройств принято различать их архитектуру и структуру.

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

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

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

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

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

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

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

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

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

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

Высокое быстродействие такой архитектуры можно получить только на задачах, в которых одинаковые вычислительные операции выполняются одновременно на различных однотипных наборах данных.

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

Что такое центральный процессор?

Центральный процессор (CPU, от англ. *Central Processing Unit*) — это основной рабочий компонент компьютера, который выполняет арифметические и логические операции, заданные программой, управляет вычислительным процессом и координирует работу всех устройств компьютера.

Центральный процессор в общем случае содержит в себе:

- арифметико-логическое устройство;
- шины данных и шины адресов;
- регистры;
- счетчики команд;
- кэш — очень быструю память малого объема (от 8 до 512 Кбайт);
- математический сопроцессор чисел с плавающей точкой.

Современные процессоры выполняются в виде *микропроцессоров*. Физически микропроцессор представляет собой *интегральную схему* — тонкую пластинку кристаллического кремния прямоугольной формы площадью всего несколько квадратных миллиметров, на которой размещены схемы, реализующие все функции процессора. Кристалл-пластинка обычно помещается в пластмассовый или керамический плоский корпус и соединяется золотыми проводками с металлическими штырьками, чтобы его можно было присоединить к системной плате компьютера.

В вычислительной системе может быть несколько параллельно работающих процессоров; такие системы называются многопроцессорными.

Основные термины

Аппаратуры (HardWare) и программного обеспечения (SoftWare),команда, система команд, функции памяти, функции процессора, принцип однородности, архитектура и структура компьютера, главные устройства компьютера.

К о н т р о л ь н ы е в о п р о с ы

1. Какова роль аппаратуры (HardWare) и программного обеспечения (SoftWare) компьютера?
2. Какие основные классы компьютеров Вам известны?
3. В чём состоит принцип действия компьютеров?
4. Из каких простейших элементов состоит программа?
5. Что такое система команд компьютера?
6. Перечислите главные устройства компьютера.
7. Опишите функции памяти и функции процессора.
8. Назовите две основные части процессора. Каково их назначение?

9. Что такое регистры? Назовите некоторые важные регистры и опишите их функции.
10. Сформулируйте общие принципы построения компьютеров.
11. В чём заключается принцип программного управления? Как выполняются команды условных и безусловных переходов?
12. В чём суть принципа однородности памяти? Какие возможности он открывает?
13. В чём заключается принцип адресности?
14. Какие архитектуры называются "фон-неймановскими"?
15. Что такое команда? Что описывает команда?
16. Какого рода информацию может содержать адресная часть команды?
17. Приведите примеры команд одноадресных, двухадресных, трёхадресных.
18. Каким образом процессор при выполнении программы осуществляет выбор очередной команды?
19. Опишите основной цикл процесса обработки команд.
20. Что понимается под архитектурой компьютера? Какие характеристики компьютера определяются этим понятием? Верно ли, что общность архитектуры разных компьютеров обеспечивает их совместимость в плане реализации функциональных элементов?
21. Что понимается под структурой компьютера? Какой уровень детализации описания компьютера может она обеспечить?
22. Перечислите распространённые компьютерные архитектуры.
23. Каковы отличительные особенности классической архитектуры?
24. Что собой представляет шина компьютера? Каковы функции общей шины (магистрали)?
25. Какую функцию выполняют контроллеры?
26. Как характер решаемых задач связан с архитектурой компьютера?
27. Какие отличительные особенности присущи многопроцессорной архитектуре? Многомашинной архитектуре? Архитектуре с параллельным процессором?
28. Что такое центральный процессор?
29. Какие основные компоненты содержат в себе современные микропроцессоры?
30. Как конструктивно выполнены современные микропроцессоры?

Тема 6. Внутренняя и внешняя память.

План:

1. Оперативная память.
2. Кэш-память.
3. Специальная память.
4. Накопители на жёстких магнитных дисках.
5. Накопители на гибких магнитных дисках.
6. Накопители на компакт-дисках.
7. Накопители на магнито-оптических компакт-дисках.
8. Накопители на магнитной ленте (стримеры) и др.

Различают два основных вида памяти — ***внутреннюю*** и ***внешнюю***.

Какие устройства образуют внутреннюю память?

В состав внутренней памяти входят *оперативная память*, *кэш-память* и *специальная память*.

· ***Оперативная память.***

Оперативная память (*ОЗУ*, англ. *RAM, Random Access Memory — память с произвольным доступом*) — это быстрое запоминающее устройство не очень большого объёма, непосредственно связанное с процессором и предназначенное для записи, считывания и хранения выполняемых программ и данных, обрабатываемых этими программами.

Оперативная память используется только *для временного хранения данных и программ*, так как, когда машина выключается, все, что находилось в ОЗУ, пропадает. Доступ к элементам оперативной памяти *прямой* — это означает, что каждый байт памяти имеет свой индивидуальный адрес.

Объем ОЗУ обычно составляет 4 - 64 Мбайта, а для эффективной работы современного программного обеспечения желательно иметь не менее 16 Мбайт ОЗУ. Обычно ОЗУ исполняется из интегральных микросхем памяти DRAM (*Dynamic RAM* — динамическое ОЗУ). Микросхемы DRAM работают медленнее, чем другие разновидности памяти, но стоят дешевле.

Каждый информационный бит в DRAM запоминается в виде электрического заряда крохотного конденсатора, образованного в структуре полупроводникового кристалла. Из-за токов утечки такие конденсаторы быстро разряжаются, и их периодически (примерно каждые 2 миллисекунды) подзаряжают специальные устройства. Этот процесс называется регенерацией памяти (Refresh Memory).

Современные микросхемы имеют ёмкость 1-16 Мбит и более. Они устанавливаются в корпуса и собираются в модули памяти.

Наиболее распространены модули типа SIMM (*Single In-Line Memory Module — модуль памяти с однорядным расположением микросхем*).

В модуле SIMM элементы памяти собраны на маленькой печатной плате длиной около 10 см. *Ёмкость* таких модулей неодинаковая — 256 Кбайт, 1, 2, 4, 8, 16, 32 и 64 Мбайта. Различные модули SIMM могут иметь разное число микросхем — девять, три или одну, и разное число контактов — 30 или 72.

Важная характеристика модулей памяти — *время доступа* к данным, которое обычно составляет 60 – 80 наносекунд.

· ***Кэш-память.***

Кэш (*англ.* ***cache***), или *сверхоперативная память* — очень быстрое ЗУ небольшого объёма, которое используется при обмене данными между микропроцессором и оперативной памятью для компенсации разницы в скорости обработки информации процессором и несколько менее быстродействующей оперативной памятью.

Кэш-памятью управляет специальное устройство — *контроллер*, который, анализируя выполняемую программу, пытается *предвидеть, какие данные и команды вероятнее всего понадобятся в ближайшее время процессору, и подкачивает их в кэш-память*. При этом возможны как "*попадания*", так и "*промахи*". В случае *попадания*, то есть, *если в кэш подкачаны нужные данные*, извлечение их из памяти происходит без задержки. Если же требуемая информация в кэше отсутствует, то процессор считывает её непосредственно из оперативной памяти. Соотношение числа попаданий и промахов определяет эффективность кэширования.

Кэш-память реализуется на микросхемах статической памяти SRAM (*Static RAM*), более быстродействующих, дорогих и малоёмких, чем DRAM.

Современные микропроцессоры имеют встроенную кэш-память, так называемый кэш первого уровня размером 8–16 Кбайт. Кроме того, на системной плате компьютера может быть установлен кэш второго уровня ёмкостью от 64 Кбайт до 256 Кбайт и выше.

· ***Специальная память.***

К устройствам специальной памяти относятся постоянная память (ROM), перепрограммируемая постоянная память (*Flash Memory*), память CMOS RAM, питаемая от батарейки, видеопамять и некоторые другие виды памяти.

Постоянная память (ПЗУ, *англ.* ROM, *Read Only Memory* — память только для чтения) — энергонезависимая память, используется для хранения данных, которые никогда не потребуют изменения. Содержание памяти специальным образом "*зашивается*" в устройстве при его изготовлении для постоянного хранения. Из ПЗУ можно только читать.

Перепрограммируемая постоянная память (*Flash Memory*) — энергонезависимая память, допускающая многократную перезапись своего содержимого с дискеты.

Прежде всего в постоянную память записывают программу управления работой самого процессора. В ПЗУ находятся программы управления дисплеем, клавиатурой, принтером, внешней памятью, программы запуска и остановки компьютера, тестирования устройств.

Важнейшая микросхема постоянной или Flash-памяти — модуль BIOS.

BIOS (*Basic Input/Output System* — базовая система ввода-вывода) — совокупность программ, предназначенных для:

- автоматического тестирования устройств после включения питания компьютера;
- загрузки операционной системы в оперативную память.

Роль BIOS двоякая: с одной стороны это неотъемлемый элемент аппаратуры (*Hardware*), а с другой строны — важный модуль любой операционной системы (*Software*).

Разновидность постоянного ЗУ — CMOS RAM.

CMOS RAM — это память с невысоким быстродействием и минимальным энергопотреблением от батарейки. Используется для хранения информации о конфигурации и составе оборудования компьютера, а также о режимах его работы.

Интегральные схемы BIOS и CMOS

Содержимое CMOS изменяется специальной программой *Setup*, находящейся в BIOS (*англ. Set-up* — устанавливать, читается "сетап").

Для хранения графической информации используется *видеопамять*.

Видеопамять (VRAM) — разновидность оперативного ЗУ, в котором хранятся закодированные изображения. Это ЗУ организовано так, что его содержимое доступно сразу двум устройствам — процессору и дисплею. Поэтому изображение на экране меняется одновременно с обновлением видеоданных в памяти.

Какие устройства образуют внешнюю память?

Внешняя память (ВЗУ) предназначена для длительного хранения программ и данных, и целостность её содержимого не зависит от того, включен или выключен компьютер. В отличие от оперативной памяти, внешняя память не имеет прямой связи с процессором. Информация от ВЗУ к процессору и наоборот циркулирует примерно по следующей цепочке:

В состав внешней памяти компьютера входят:

- накопители на *жёстких* магнитных дисках;
- накопители на *гибких* магнитных дисках;
- накопители на *компакт-дисках*;
- накопители на *магнито-оптических* компакт-дисках;
- накопители на *магнитной ленте* (стримеры) и др.

Накопители на гибких магнитных дисках

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

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

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

Информация записывается по концентрическим *дорожкам* (*трекам*), которые делятся на *секторы*. Количество дорожек и секторов зависит от типа и формата дискеты. Сектор хранит минимальную порцию информации, которая может быть записана на диск или считана. Ёмкость сектора постоянна и составляет 512 байтов.

На дискете можно хранить от 360 Килобайт до 2,88 Мегабайт информации.

В настоящее время наибольшее распространение получили дискеты со следующими характеристиками: диаметр 3,5 дюйма (89 мм), ёмкость 1,44 Мбайт, число дорожек 80, количество секторов на дорожках 18.

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

Накопитель связан с процессором через *контроллер гибких дисков*.

Накопители на жестких магнитных дисках

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

Накопитель на жёстких магнитных дисках (*англ*. HDD — *Hard Disk Drive*) или *винчестерский накопитель* — это наиболее массовое запоминающее устройство большой ёмкости, в котором носителями информации являются круглые алюминиевые пластины — *плоттеры*, обе поверхности которых покрыты слоем магнитного материала. Используется для постоянного хранения информации — программ и данных.

Как и у дискеты, рабочие поверхности плоттеров разделены на кольцевые концентрические дорожки, а дорожки — на секторы. Головки считывания-записи вместе с их несущей конструкцией и дисками заключены в герметически закрытый корпус, называемый *модулем данных*. При установке модуля данных на дисковод он автоматически соединяется с системой, подкачивающей очищенный охлажденный воздух.

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

Винчестерские накопители имеют *очень большую ёмкость*: от сотен Мегабайт до десятков Гбайт. У современных моделей скорость вращения шпинделя достигает 7200 оборотов в минуту, среднее время поиска данных — 10 мс, максимальная скорость передачи данных до 40 Мбайт/с.
В отличие от дискеты, винчестерский диск *вращается непрерывно*.

Винчестерский накопитель связан с процессором через *контроллер жесткого диска*.

В се современные накопители снабжаются *встроенным «кэш»ем* (64 Кбайт и более), который существенно повышает их производительность.

Накопители на компакт-дисках

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

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

Достоинства CD-ROM:

- При малых физических размерах CD-ROM обладают *высокой информационной ёмкостью*, что позволяет использовать их в справочных системах и в учебных комплексах с богатым иллюстративным материалом; *один CD, имея размеры примерно дискеты, по информационному объёму равен почти 500 таким дискетам*;
- *Считывание информации с CD происходит с высокой скоростью*, сравнимой со скоростью работы винчестера;
- CD просты и удобны в работе, практически не изнашиваются;
- CD не могут быть поражены вирусами;
- На CD-ROM *невозможно случайно стереть информацию*;
- Стоимость хранения данных (в расчете на 1 Мбайт) низкая.

В отличие от магнитных дисков, компакт-диски имеют не множество кольцевых дорожек, а *одну — спиральную*, как у грампластинок. В связи с этим, угловая скорость вращения диска не постоянна. Она линейно уменьшается в процессе продвижения читающей магнитной головки к центру диска.

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

Что такое накопитель CD-ROM с технической точки зрения?

Участки CD, на которых записаны символы "0" и "1", отличаются коэффициентом отражения лазерного луча, посылаемого накопителем CD-ROM. Эти отличия улавливаются фотоэлементом, и общий сигнал преобразуется в соответствующую последовательность нулей и единиц.

Многие накопители CD-ROM способны воспроизводить обычные аудио-CD. Это позволяет пользователю, работающему за компьютером, слушать музыку в фоновом режиме.

Со временем на смену CD-ROM могут прийти *цифровые видеодиски DVD*(читается "ди-ви-ди"). Эти диски имеют тот же размер, что и обычные CD, но вмещают 4,7 Гбайт данных, т.е. по объёму заменяют *семь стандартных дисков CD-ROM*. В скором времени ёмкость дисков DVD возрастет до 17 Гбайт. На таких дисках будут выпускаться полноэкранные видеофильмы отличного качества, программы-тренажёры, мультимедийные игры и многое другое.

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

Записывающие оптические и магнитооптические накопители

- Накопитель на магнито-оптических компакт-дисках CD-MO (Compact Disk-Magneto Optical). Диски CD-MO можно многократно использовать для записи, но они не читаются на традиционных дисководах CD-ROM. Ёмкость от 128 Мбайт до 2,6 Гбайт.
- Записывающий накопитель CD-R (Compact Disk Recordable) способен, наряду с прочтением обычных компакт-дисков, записывать информацию на специальные оптические диски. Ёмкость 650 Мбайт.
- Накопитель WARM (Write And Read Many times), позволяет производить многократную запись и считывание.
- Накопитель WORM (Write Once, Read Many times), позволяет производить однократную запись и многократное считывание.

Накопители на магнитной ленте (стримеры) и накопители на сменных дисках

Стример (англ. *tape streamer*) — устройство для резервного копирования больших объёмов информации. В качестве носителя здесь применяются кассеты с магнитной лентой ёмкостью 1 - 2 Гбайта и больше.

Стримеры позволяют записать на небольшую кассету с магнитной лентой огромное количество информации. Встроенные в стример средства аппаратного сжатия позволяют автоматически уплотнять информацию перед её записью и восстанавливать после считывания, что увеличивает объём сохраняемой информации.

Недостатком стримеров является их сравнительно *низкая скорость* записи, поиска и считывания информации.

В последнее время всё шире используются накопители на сменных дисках, которые позволяют не только увеличивать объём хранимой информации, но и переносить информацию между компьютерами. Объём сменных дисков — от сотен Мбайт до нескольких Гигабайт.

Основные термины

Оперативная память, Кеш память, специальная память, ОЗУ, статическая и динамическая память, внешняя память, гибкий магнитный диск, жёсткие магнитные диски.

К о н т р о л ь н ы е в о п р о с ы

1. Перечислите основные компоненты внутренней памяти.
2. Что представляет собой ОЗУ? Каково её назначение?
3. В чём разница между памятью статической и динамической?
4. Что собой представляет модуль памяти типа SIMM? Какие другие типы модулей памяти Вы знаете?
5. Каково назначение кэш-памяти? Каким образом она реализуется?
6. Что такое специальная память? Характеризуйте её основные виды.
7. Что такое BIOS и какова её роль?
8. Каково назначение внешней памяти? Перечислите разновидности устройств внешней памяти.
9. Что собой представляет гибкий диск?
10. Как работают накопители на гибких магнитных дисках и накопители на жёстких магнитных дисках?
11. Каковы достоинства и недостатки накопителей на компакт-дисках?
12. Опишите работу стримера.

Тема 7. Программное обеспечение компьютеров

План:

1. Программное обеспечение. Классификация программного обеспечения.
2. Прикладные программы.
3. Системные программы.
4. Операционная система.
5. Файловая система.
6. Структура операционной системы MS DOS?
7. Что такое программы-оболочки?

Работа на ЭВМ обычно проходит в форме диалога человека с компьютером. Человек просматривает информацию на экране компьютера, указывает на нее мышкой, нажимает клавиши, набирает команды, вводит слова, числа, фразы и т. п. В ответ компьютер выводит свою информацию: сообщения, меню, заставки, диаграммы, рисунки, результаты вычислений и обработки данных.

Работа ЭВМ основана на использовании программ. **Программы для ЭВМ** - это форма представления данных и команд, предназначенных для получения определенных результатов или способа функционирования ЭВМ.

Совокупность программ для данного типа ЭВМ определяет все многообразие их применений. На персональных компьютерах наиболее часто применяются игры, редакторы текстов, базы данных, информационные системы, электронные таблицы, системы программирования и т. п.

Главной среди программ на ЭВМ является **операционная система,** которая постоянно хранится в долговременной памяти компьютера. Работа ЭВМ начинается с загрузки операционной системы, а все остальные программы запускаются с помощью операционной системы.

Операционная система - это главная программа, управляющая работой компьютера в целом. На персональных компьютерах типа IBM PC используются в основном операционные системы **MS DOS** и **Windows.** В персональных компьютерах Macintosh применяется операционная система OS/7.

Операционная система **MS DOS** - это самая простая операционная система для компьютеров IBM PC. Она используется на всех младших моделях IBM PC и может применяться на всех старших моделях компьютеров этого же типа.

Операционная система **Windows** - наиболее современная и удобная операционная система для старших моделей персональных компьютеров IBM PC. Эта система может использоваться только на компьютерах старших моделей с оперативной памятью более 2 Мбайт и памятью на жестких дисках не менее 80 Мбайт.

На персональных компьютерах IBM PC используются несколько версий операционной системы Windows; созданных всемирно известной фирмой Microsoft: Windows 3.1, Windows-95, Windows-98, Windows-2000, отличающихся своими функциями и возможностями.

Основными объектами во всех операционных системах на ЭВМ являются файлы, программы и каталоги. Все программы в ЭВМ представляются отдельными файлами или наборами файлов, хранящихся в определенном каталоге.

Файлы - это последовательность записей на машинных носителях - магнитных или оптических дисках, магнитных или перфолентах и т.п. Все данные и программы на ЭВМ записываются в виде файлов или наборов файлов. Все файлы в памяти ЭВМ имеют уникальные имена.

Совокупности файлов в памяти ЭВМ объединяются в форме каталогов и подкаталогов. Каждый каталог имеет свое уникальное имя. Имя подкаталога образуется из его собственного имени и имени каталога, в котором он находится. Имена каталогов (оглавлений) записываются большими (прописными) буквами, а имена файлов - малыми (строчными) буквами.

В операционных системах MS DOS и Windows **имена файлов** образуются из латинских букв и цифр с добавлением трехбуквенных окончаний после точки. Для записи окончаний в этих операционных системах приняты правила:

.exe - программа, готовая к выполнению;

.com - программа, готовая к выполнению;

.bat - командный файл операционной системы;

.txt - текстовый файл;

.doc - текстовый файл.

Работа с любыми операционными системами - это в основном работа над каталогами файлов и программ, размещенных на магнитных и оптических дисках. Эта работа состоит в просмотре каталогов и подкаталогов, копировании файлов и запуске тех или иных программ.

В любой современной операционной системе работа с ЭВМ происходит в основном с помощью менеджеров программ и файлов. Эта программа позволяет человеку в диалоге с компьютером просматривать каталоги программ и файлов во внешней памяти:

C: A:

Name	Name	Name	Name	Name	Name
...	begin.bat	file2.dat	TUTOR	MUSIC	
DOS	prog1.exe		TEACHER	GRAF	
NC	prog2.exe		CALC	DOC	
KAYMIN	text1.txt		EDIT		
BOOK	text2.txt		BASE		
PACET	file1.dat		PROLOG		
text1.txt	4096	26.12.96	6720 bytes in 1 files selected		

В приведенном примере указаны два **каталога:** каталог на жестком диске C, на котором размещена операционная система DOS, и каталог на гибком диске A с пакетом программ, включающим электронный учебник TEACHER, клавиатурный тренажер TUTOR и другие учебные программы из пакета программ для лабораторных работ по информатике.

Что такое программное обеспечение?

Под *программным обеспечением* (Software) понимается совокупность программ, выполняемых вычислительной системой.

К программному обеспечению (ПО) относится также вся область деятельности по проектированию и разработке ПО:

- технология проектирования программ (например, нисходящее проектирование, структурное и объектно-ориентированное проектирование и др.);
- методы тестирования программ [ссылка, ссылка];
- методы доказательства правильности программ;
- анализ качества работы программ;
- документирование программ;
- разработка и использование программных средств, облегчающих процесс проектирования программного обеспечения, и многое другое.

Программное обеспечение – *неотъемлемая часть компьютерной системы.* Оно является логическим родолжением технических средств. Сфера применения конкретного компьютера определяется созданным для него ПО.

Сам по себе компьютер не обладает знаниями ни в одной области применения. Все эти знания сосредоточены в выполняемых на компьютерах программах.

Программное обеспечение современных компьютеров включает миллионы программ — от игровых до научных.

Как классифицируется программное обеспечение?

В первом приближении все программы, работающие на компьютере, можно условно разделить на три категории :

прикладные программы, непосредственно обеспечивающие выполнение необходимых пользователям работ;

системные программы, выполняющие различные вспомогательные функции, например:

- управление ресурсами компьютера;
- создание копий используемой информации;
- проверка работоспособности устройств компьютера;
- выдача справочной информации о компьютере и др.;

инструментальные программные системы, облегчающие процесс создания новых программ для компьютера.

При построении классификации ПО нужно учитывать тот факт, что стремительное развитие вычислительной техники и расширение сферы приложения компьютеров резко ускорили процесс эволюции программного обеспечения.

Если раньше можно было по пальцам перечислить основные категории ПО — операционные системы, трансляторы, пакеты прикладных программ, то сейчас ситуация коренным образом изменилась.

Развитие ПО пошло как вглубь (появились новые подходы к построению операционных систем, языков программирования и т.д.), так и вширь (прикладные программы перестали быть прикладными и приобрели самостоятельную ценность).

Соотношение между требующимися программными продуктами и имеющимися на рынке меняется очень быстро. Даже классические программные продукты, такие, как операционные системы, непрерывно развиваются и наделяются интеллектуальными функциями, многие из которых ранее относились только к интеллектуальным возможностям человека.

Кроме того, появились нетрадиционные программы, классифицировать которые по устоявшимся критериям очень трудно, а то и просто невозможно, как, например, программа — *электронный собеседник.*

На сегодняшний день можно сказать, что более или менее определённо сложились следующие группы программного обеспечения:

- операционные системы и оболочки;
- системы программирования (трансляторы, библиотеки подпрограмм, отладчики и т.д.);
- инструментальные системы;
- интегрированные пакеты программ;
- динамические электронные таблицы;
- системы машинной графики;
- системы управления базами данных (СУБД);
- прикладное программное обеспечение.

Разумеется, эту классификацию нельзя считать исчерпывающей, но она более или менее наглядно отражает направления совершенствования и развития программного обеспечения.

Какие программы называют прикладными?

Прикладная программа — это любая конкретная программа, способствующая решению какой-либо задачи в пределах данной проблемной области.

Например, там, где на компьютер возложена задача контроля за финансовой деятельностью какой-либо фирмы, прикладной будет программа подготовки платежных ведомостей.

Прикладные программы могут носить и общий характер, например, обеспечивать составление и печатание документов и т.п.

В противоположность этому, операционная система или инструментальное ПО не вносят прямого вклада в удовлетворение конечных потребностей пользователя.

Прикладные программы могут использоваться либо автономно, то есть решать поставленную задачу без помощи других программ, либо в составе программных комплексов или пакетов.

Какова роль и назначение системных программ?

Системные программы выполняются вместе с прикладными и служат для управления ресурсами компьютера — центральным процессором, памятью, вводом-выводом.

Это программы общего пользования, которые предназначены для всех пользователей компьютера. Системное программное обеспечение разрабатывается так, чтобы компьютер мог эффективно выполнять прикладные программы.

Среди десятков тысяч системных программ особое место занимают операционные системы, которые обеспечивают управление ресурсами компьютера с целью их эффективного использования.

Важными классами системных программ являются также программы вспомогательного назначения — утилиты (лат. *utilitas* — польза). Они либо расширяют и дополняют соответствующие возможности операционной системы, либо решают самостоятельные важные задачи. Кратко опишем некоторые разновидности утилит:

- ***программы контроля, тестирования и диагностики***, которые используются для проверки правильности функционирования устройств компьютера и для обнаружения неисправностей в процессе эксплуатации; указывают причину и место неисправности;
- ***программы-драйверы,*** которые расширяют возможности операционной системы по управлению устройствами ввода-вывода, оперативной памятью и т.д.; с помощью драйверов возможно подключение к компьютеру новых устройств или нестандартное использование имеющихся;
- ***программы-упаковщики (архиваторы)***, которые позволяют записывать информацию на дисках более плотно, а также объединять копии нескольких файлов в один архивный файл;
- ***антивирусные программы,*** предназначенные для предотвращения заражения компьютерными вирусами и ликвидации последствий заражения вирусами;

Компьютерный вирус — это специально написанная небольшая по размерам программа, которая может "приписывать" себя к другим программам для выполнения каких-либо вредных действий — портит файлы, "засоряет" оперативную память и т.д.

- ***программы оптимизации и контроля качества дискового пространства;***
- **программы восстановления информации, форматирования, защиты данных;**
- ***коммуникационные программы,*** организующие обмен информацией между компьютерами;
- ***программы для управления памятью,*** обеспечивающие более гибкое использование оперативной памяти;
- ***программы для записи*** CD-ROM, CD-R и многие другие.

Часть утилит входит в состав операционной системы, а другая часть функционирует независимо от нее, т.е. автономно.

Что такое операционная система?

Операционная система — это комплекс взаимосвязанных системных программ, назначение которого — организовать взаимодействие пользователя с компьютером и выполнение всех других программ.

Операционная система выполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.

Операционная система обычно хранится во внешней памяти компьютера — *на диске*. При включении компьютера она считывается с дисковой памяти и размещается в *ОЗУ*.

Этот процесс называется *загрузкой операционной системы*.

В функции операционной системы входит:

- осуществление диалога с пользователем;
- ввод-вывод и управление данными;
- планирование и организация процесса обработки программ;
- распределение ресурсов (оперативной памяти и кэша, процессора, внешних устройств);
- запуск программ на выполнение;
- всевозможные вспомогательные операции обслуживания;
- передача информации между различными внутренними устройствами;
- программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).

Операционную систему можно назвать программным продолжением устройства управления компьютера. Операционная система скрывает от пользователя сложные ненужные подробности взаимодействия с аппаратурой, образуя прослойку между ними. В результате этого люди освобождаются от очень трудоёмкой работы по организации взаимодействия с аппаратурой компьютера.

В зависимости от количества одновременно обрабатываемых задач и числа пользователей, которых могут обслуживать ОС, ***различают четыре основных класса операционных систем:***

1. *однопользовательские однозадачные*, которые поддерживают одну клавиатуру и могут работать только с одной (в данный момент) задачей;
2. *однопользовательские однозадачные с фоновой печатью*, которые позволяют помимо основной задачи запускать одну дополнительную задачу, ориентированную, как правило, на вывод

информации на печать. Это ускоряет работу при выдаче больших объёмов информации на печать;

3. *однопользовательские многозадачные*, которые обеспечивают одному пользователю параллельную обработку нескольких задач. Например, к одному компьютеру можно подключить несколько принтеров, каждый из которых будет работать на "свою" задачу;
4. *многопользовательские многозадачные*, позволяющие на одном компьютере запускать несколько задач нескольким пользователям. Эти ОС очень сложны и требуют значительных машинных ресурсов.

В различных моделях компьютеров используют операционные системы с разной архитектурой и возможностями. Для их работы требуются разные ресурсы. Они предоставляют разную степень сервиса для программирования и работы с готовыми программами.

Операционная система для персонального компьютера, ориентированного на профессиональное применение, должна содержать следующие основные компоненты:

- Ø программы управления вводом/выводом;
- Ø программы, управляющие файловой системой и планирующие задания для компьютера;
- Ø процессор командного языка, который принимает, анализирует и выполняет команды, адресованные операционной системе.

Каждая операционная система имеет свой командный язык, который позволяет пользователю выполнять те или иные действия:

- ü обращаться к каталогу;
- ü выполнять разметку внешних носителей;
- ü запускать программы;
- ü ... другие действия.

Анализ и исполнение команд пользователя, включая загрузку готовых программ из файлов в оперативную память и их запуск, осуществляет командный процессор операционной системы.

Для управления внешними устройствами компьютера используются специальные системные программы — драйверы. Драйверы стандартных устройств образуют в совокупности базовую систему ввода-вывода (BIOS), которая обычно заносится в постоянное ЗУ компьютера.

Что такое файловая система ОС?

Файл (англ.*file*,папка) — это место постоянного хранения информации: программ, данных для их работы, текстов, закодированных изображений, звуков и др.

Файловая система — это средство для организации хранения файлов на каком-либо носителе.

Файлы физически реализуются как *участки памяти на внешних носителях* — магнитных дисках или CD-ROM.

Каждый файл занимает некоторое количество блоков дисковой памяти. Обычная длина блока — 512 байт.

Обслуживает файлы специальный модуль операционной системы, называемый *драйвером файловой системы*. Каждый файл имеет имя, зарегистрированное в *каталоге* — оглавлении файлов.

Каталог (иногда называется *директорией* или *папкой*) доступен пользователю через командный язык операционной системы.

Его можно просматривать, переименовывать зарегистрированные в нем файлы, переносить их содержимое на новое место и удалять.

Каталог может иметь собственное имя и храниться в другом каталоге наряду с обычными файлами: так образуются иерархические файловые структуры.

Что происходит, когда пользователь подает операционной системе команду "*открыть файл ...*", в которой указано *имя файла* и *имя каталога*, в котором размещён этот файл?

Для выполнения этой команды драйвер файловой системы обращается к своему *справочнику*, выясняет, какие блоки диска соответствуют указанному файлу, а затем передает запрос на считывание этих блоков драйверу диска.

При выполнении команды "*сохранить файл*" драйвер файловой системы ищет на диске незанятые блоки, отмечает их, как распределённые для вновь созданного файла, и передаёт драйверу диска запрос на запись в эти блоки данных пользователя.

Драйвер файловой системы обеспечивает доступ к информации, записанной на магнитный диск, по имени файла и распределяет пространство на магнитном диске между файлами.

Для выполнения этих функций драйвер файловой системы хранит на диске не только информацию пользователя, но и свою собственную служебную информацию. *В служебных областях диска хранится список всех файлов и каталогов*, а также различные дополнительные справочные таблицы, служащие для повышения скорости работы драйвера файловой системы.

К файловой системе имеет доступ также и любая прикладная программа, для чего во всех языках программирования имеются специальные процедуры.

Понятие файла может быть обращено на любой источник или потребитель информации в машине, например, в качестве файла для программы могут выступать *принтер, дисплей, клавиатура* и др.

Структура файловой системы и структура хранения данных на внешних магнитных носителях определяет удобство работы пользователя, скорость доступа к файлам и т.д.

Какова структура операционной системы MS DOS?

Операционная система MS DOS (Microsoft Disk Operating System) — самая распространенная ОС на 16-разрядных персональных компьютерах. Она состоит из следующих основных модулей:

- базовая система ввода/вывода (BIOS);
- блок начальной загрузки (Boot Record);
- модуль расширения базовой системы ввода/вывода (IO.SYS);
- модуль обработки прерываний (MSDOS.SYS);
- командный процессор (COMMAND.COM);
- утилиты MS DOS.

Каждый из указанных модулей выполняет определенную часть функций, возложенных на ОС. Места постоянного размещения этих модулей различны. Так, *базовая система ввода/вывода находится в постоянном запоминающем устройстве* (ПЗУ), а не на дисках, как все остальные модули.

Базовая система ввода/вывода (BIOS) выполняет наиболее простые и универсальные услуги операционной системы, связанные с осуществлением *ввода-вывода*. В функции BIOS входит также *автоматическое тестирование основных аппаратных компонентов* (оперативной памяти и др.) при включении машины и *вызов блока начальной загрузки DOS*.

Блок начальной загрузки (или просто *загрузчик*) — это очень короткая программа, единственная функция которой заключается в считывании с диска в оперативную память двух других частей DOS — модуля расширения базовой системы ввода/вывода и модуля обработки прерываний.

Модуль расширения базовой системы ввода/вывода дает возможность использования *дополнительных драйверов*, обслуживающих новые внешние устройства, а также *драйверов для нестандартного обслуживания внешних устройств*.

Модуль обработки прерываний реализует основные высокоуровневые услуги DOS, поэтому его и называют основным.

Командный процессор DOS обрабатывает команды, вводимые пользователем.

Утилиты DOS — это программы, поставляемые вместе с операционной системой в виде отдельных файлов. Они выполняют действия обслуживающего характера, например, разметку дискет, проверку дисков и т.д.

Что такое программы-оболочки?

Оболочки — это программы, созданные для упрощения работы со сложными программными системами, такими, например, как DOS. Они преобразуют неудобный командный пользовательский интерфейс в дружественный графический интерфейс или интерфейс типа "меню". Оболочки предоставляют пользователю удобный доступ к файлам и обширные сервисные услуги.

Самая популярная у пользователей IBM-совместимого ПК оболочка — пакет программ *Norton Commander*. Он обеспечивает:

- создание, копирование, пересылку, переименование, удаление, поиск файлов, а также изменение их атрибутов;
- отображение дерева каталогов и характеристик входящих в них файлов в форме, удобной для восприятия человека;
- создание, обновление и распаковку архивов (групп сжатых файлов);
- просмотр текстовых файлов;
- редактирование текстовых файлов;

- выполнение из её среды практически всех команд DOS;
- запуск программ;
- выдачу информации о ресурсах компьютера;
- создание и удаление каталогов;
- поддержку межкомпьютерной связи;
- поддержку электронной почты через модем.

В начале 90-х годов во всем мире огромную популярность приобрела графическая оболочка *MS-Windows 3.x*, преимущество которой состоит в том, что она облегчает использование компьютера, и её графический интерфейс вместо набора сложных команд с клавиатуры позволяет выбирать их мышью из меню практически мгновенно. Операционная среда Windows, работающая совместно с операционной системой DOS, реализует все свойства, необходимые для производительной работы пользователя, в том числе – многозадачный режим.

Оболочка Norton Navigator — это набор мощных программ для управления файлами, расширяющий возможности Windows. Позволяет экономить время практически на всех операциях: поиск файлов, копирование и перемещение файлов, открытие каталогов.

Что такое транслятор, компилятор, интерпретатор?

Транслятор (англ. *translator* — переводчик) — это программа-переводчик. Она преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд.

Трансляторы реализуются в виде компиляторов или интерпретаторов. С точки зрения выполнения работы компилятор и интерпретатор существенно различаются.

Компилятор (англ. *compiler* — составитель, собиратель) читает всю программу *целиком*, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется.

Интерпретатор (англ. *interpreter* — истолкователь, устный переводчик) переводит и выполняет программу *строка за строкой*.

После того, как программа откомпилирована, ни сама исходная программа, ни компилятор более не нужны. В то же время программа, обрабатываемая интерпретатором, должна заново *переводиться* на машинный язык при каждом очередном запуске программы.

Откомпилированные программы работают быстрее, но *интерпретируемые* проще исправлять и изменять.

Каждый конкретный язык ориентирован либо на компиляцию, либо на интерпретацию — в зависимости от того, для каких целей он создавался. Например, *Паскаль* обычно используется для решения довольно сложных задач, в которых важна скорость работы программ. Поэтому данный язык обычно реализуется с помощью *компилятора*.

С другой стороны, *Бейсик* создавался как язык для начинающих программистов, для которых построчное выполнение программы имеет неоспоримые преимущества.

Иногда для одного языка имеется *и компилятор, и интерпретатор*. В этом случае для разработки и тестирования программы можно воспользоваться интерпретатором, а затем откомпилировать отлаженную программу, чтобы повысить скорость ее выполнения.

Что такое системы программирования?

Система программирования — это система для разработки новых программ на конкретном языке программирования.

Современные системы программирования обычно предоставляют пользователям мощные и удобные средства разработки программ. В них входят:

- компилятор или интерпретатор;
- интегрированная среда разработки;
- средства создания и редактирования текстов программ;
- обширные библиотеки стандартных программ и функций;
- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;
- "дружественная" к пользователю диалоговая среда;
- многооконный режим работы;
- мощные графические библиотеки; утилиты для работы с библиотеками

- встроенный ассемблер;
- встроенная справочная служба;
- другие специфические особенности.

Популярные системы программирования – *Turbo Basic*, *Quick Basic*, *Turbo Pascal*, *Turbo C*.

В последнее время получили распространение системы программирования, ориентированные на создание *Windows-приложений*:

Borland Delphi 3.0

- пакет *Borland Delphi* (Дельфи) — блестящий наследник семейства компиляторов Borland Pascal, предоставляющий качественные и очень удобные средства визуальной разработки. Его исключительно быстрый компилятор позволяет эффективно и быстро решать практически любые задачи прикладного программирования.
- пакет *Microsoft Visual Basic* — удобный и популярный инструмент для создания Windows-программ с использованием визуальных средств. Содержит инструментарий для создания *диаграмм* и *презентаций*.
- пакет *Borland C++* — одно из самых распространённых средств для разработки DOS и Windows приложений.

Основные термины

Программное обеспечение, прикладные программы, системные программы, инструментальные программы, начальная загрузка операционной системы, файл, файловая система, командный и графический интерфейс.

К о н т р о л ь н ы е в о п р о с ы

1. Что такое программа?
2. Что включает в себя понятие "программное обеспечение"?
3. Назовите и характеризуйте основные категории программного обеспечения.
4. В чем отличие прикладных программ от системных и инструментальных?
5. Что входит в системное программное обеспечение?
6. В чем состоит назначение операционной системы?
7. Характеризуйте основные классы операционных систем.
8. Опишите процесс начальной загрузки операционной системы в оперативную память компьютера.
9. Что такое файл?
10. Как организована файловая система?
11. Какой модуль операционной системы осуществляет обслуживание файлов?
12. Приведите пример иерархической файловой структуры.
13. Что такое базовая система ввода-вывода (BIOS), и в каком разделе памяти она размещается?
14. Из каких основных модулей состоит операционная система MS-DOS?
15. Назовите основные разновидности программ-утилит и дайте им краткую характеристику.
16. К каким категориям программного обеспечения относятся программные пакеты:
    Norton Commander;
    MS-DOS;
    Windows 3.x;
    Windows-NT, Windows 95;
    Microsoft Word;
    Adobe PageMaker;
    Turbo Pascal, Turbo Basic;
    Microsoft Excel, Lotus;
    FoxPro, Access for Windows;
    Microsoft Office, Microsoft Works?
17. Для чего предназначен пакет программ Norton Commander?

18. Какой вид интерфейса удобнее для пользователя — командный или графический?
19. Чем объясняется широкая популярность пакета Norton Commander?
20. Что такое компьютерные вирусы, в чем состоят их вредные действия?
21. Какие существуют средства борьбы с компьютерными вирусами?
22. В чем суть процесса сжатия информации?
23. В чем отличие процесса интерпретации от процесса компиляции?

Тема 8. Характеристика, функции и области применения операционных систем Windows ME, Windows NT, Windows 95, Windows XP и Windows 2000. План:

1. Операционные системы Windows NT и Windows 95.
2. Характеристика ОС Windows 3.1.
3. Характеристика ОС Windows 98/98 SE.
4. Характеристика ОС Windows 2000.
5. Характеристика ОС Windows ME (Millennium Edition).
6. Характеристика ОС Windows XP.

Известно, что операционная система (ОС) – это самая главная программа, позволяющая общение человека с компьютером. Среди обязательных частей ОС следует упомянуть **о ядре** (командный интерпретатор) – «переводчика» с программного языка на язык машинных кодак, **о драйверах** - специальные программы для управления различными устройствами компьютера **об интерфейсе** – оболочка, с которой общается пользователь. На сегодня для любой ОС **графический интерфейс** является неизменным атрибутом.

При анализе ОС следует особо обратить внимание на следующие критерии:

1. **однозадачные** (DOS), выполняющие в одно и тоже время не более одной задачи, и **многозадачные** (Windows 98/ME), способные одновременно управляться с несколькими процессами, деля между ними мощность компьютера;
2. **однопользовательские** (для обслуживания одного клиента Windows 98/ME) и **многопользовательские** (рассчитанные для работы с группой пользователей одновременно - Windows NT/2000). Для домашнего использования лучше использовать однопользовательские ОС;
3. **разрядность**

Сейчас появились 64 разрядные ОС, в перспектива ожидается 128 – разрядные ОС.
Специализация, предназначение для той или иной ОС. Универсальных ОС не существует. Одни более пригодны для работы в сети, других выбирают программисты, третьих используют домашние пользователи. Значит, знание и использование одной ОС на сегодняшний день не достаточно.
Проанализируем ОС корпорации Microsoft по этим критериям.

MS DOS

16- разрядная однозадачная обладала «Интерфейсом командной строки», графика, сервис почти отсутствовал. Хотя ОС Novell DOS зарекомендовала себя как превосходная сетевая ОС, обладающая более лучшими сервисными возможностями.
Основными недостатками MS DOS являлись работа с оперативной памятью (640 Кбайт ОП), невозможность работы в графическом режиме полноценно, однозадачность ОС.

Хотя с появлением MS Windows 95, DOS практически сошла со сцены, она установлена на компьютерах в качестве составляющей ядра Windows. Даже в 1999 году фирма IBM выпустила новую версия – DOS 2000.

Что собой представляют операционные системы Windows NT и Windows 95?

Windows NT (*NT* — англ. *New Technology*) — это операционная система, а не просто графическая оболочка. Она использует все возможности новейших моделей персональных компьютеров и работает без DOS.

Windows NT — 32-разрядная ОС со встроенной сетевой поддержкой и развитыми многопользовательскими средствами. Она предоставляет пользователям истинную многозадачность, многопроцессорную поддержку, секретность, защиту данных и многое другое.

Эта операционная система очень удобна для пользователей, работающих в рамках локальной сети, для коллективных пользователей, особенно для групп, работающих над большими проектами и обменивающихся данными.

Windows 95 представляет собой универсальную высокопроизводительную многозадачную и многопотоковую 32-разрядную ОС нового поколения с графическим интерфейсом и расширенными сетевыми возможностями.

Windows 95 — интегрированная среда, обеспечивающая эффективный обмен информацией между отдельными программами и предоставляющая пользователю широкие возможности работы с мультимедиа, обработки текстовой, графической. звуковой и видеоинформации.

Интегрированность подразумевает также *совместное использование ресурсов компьютера всеми программами*.

Эта операционная система *обеспечивает работу пользователя в сети*, предоставляя встроенные средства поддержки для обмена файлами и меры по их защите, возможность совместного использования принтеров, факсов и других общих ресурсов. Windows 95 позволяет отправлять сообщения электронной почтой, факсимильной связью, поддерживает удаленный доступ.

Применяемый в *Windows 95* защищённый режим не позволяет прикладной программе в случае сбоя нарушить работоспособность системы, надежно предохраняет приложения от случайного вмешательства одного процесса в другой, обеспечивает определённую устойчивость к вирусам.

Пользовательский интерфейс *Windows 95* прост и удобен.

В отличие от оболочки *Windows 3* эта операционная система не нуждается в установке на компьютере операционной системы DOS. Она предназначена для установки на настольных ПК и компьютерах блокнотного типа с процессором 486 или Pentium.

Рекомендуемый размер оперативной памяти 8-16 Мбайт.

После включения компьютера и выполнения тестовых программ BIOS операционная система *Windows 95* автоматически загружается с жесткого диска. После загрузки и инициализации системы на экране появляется *рабочий стол*, на котором размещены различные *графические объекты*. Пользовательский интерфейс спроектирован так, чтобы максимально облегчить усвоение этой операционной системы новичками и создать комфортные условия для пользователя.

Windows 3.1.

Эта ОС фактически представляла из себя лишь графическую оболочку, настройку над уставленным по компьютере компонентом MS DOS. Из достоинств этой ОС можно показать полноценный графический интерфейс, примитивная (2-3 приложения) многозадачность, использование всей установленной оперативной памяти. Хотя **Windows 3.1. и** отличалась своей редкостной неустойчивостью, частыми «зависаниями» и большим количеством ошибок, к середине 90-х годов большинство компьютеров были оснащены этой ОС.

Windows NT (Windows New Technology).

32-разрядная (первая версия – 1993 год, последняя – 1998 год), стабильная, надежная система, случаи ошибок, «зависаний» встречаются очень редко. Это происходит потому, что эта ОС каждой программе выделяет свою долю адресного пространства оперативной памяти и системных ресурсов. Следует подчеркнуть, что большая часть достоинств этой ОС проявляется лишь в сетевом режиме работы.
Сегодня под управлением **Windows NT** и ее преемницы **Windows 2000** работает большинство рабочих станций и серверов в крупных локальных сетях.

Windows 95.

Хотя в качестве основы в **Windows 95** использовалась старая DOS считается, что впервые **Windows** превратилась из графической настройки для DOS в полноценную ОС, это позволяло не терять возможности работать в DOS – режиме и не расставаться с привычными DOS – программами.

Windows 95 – 16 – 32 разрядная ОС. Выпушенные новые 32 – разрядные версии настоящей ОС используют в полной мере возможности современных процессоров, а старые 16-разрядные позволяют работать с новой ОС без всяких проблем. Использованные 16-разрядных модулей в ОС **Windows 95** оставляет возможность унаследования шаткость и нестабильность своих предшественников, зато можно гордиться новым графическим интерфейсом более элегантным, удобным и просто красивым, из какой ОС они не были бы заимствованы.

В новой версии **Windows 95**, вышедшей летом 1996 года OSR2 (OEM Service Release) пользователем вместо файловой системы FAT 16 могут использовать файловую систему FAT 32, что экономит место на диске.

Windows 98/98 SE.

Говоря по большому, **Windows 98** отличается от своих предшественников не так уж и много. Изменения коснулись интерфейса – «Рабочий Стол» стал еще красивее, он полностью интегрирован со сферой Internet. Стерта разница между файлами и папками на персональном компьютере и объектами Всемирной Информационной Паутины (WWW). В обеих случаях средство работы – это программа Internet Explorer. Другое отличие расширенные возможности управления интерфейсам – с помощью встроенных средств предлагаемой сервис стал помимо богаче.

В новой версии **Windows 98 - Windows 98 SE,** вышедший в конце 1999 года имеются незначительные отличие - в ее составе пятая версия Internet Explorer, обновленная системе соединения с Internet новая система исправления ошибок и новая библиотека драйверов устройств.

Windows 2000.

Windows 2000 должна была стать стандартом не только для «корпоративного» рынка, но и для домашних «персоналок». **Windows 2000** унаследовал стабильное, полностью 32 – разрядное ядро **Windows NT** и удобную оболочку от **Windows 98.** Унаследовав защищенность, отличные сетевые возможности и сервисы NT, **Windows 2000** стала удобней и дружественной «домашнему» пользователю.

Как и **Windows NT, Windows 2000** выпущена в сервисном варианте для установки на главный, управляющей компьютер сети, клиентском (Professional) – для рабочих станций. Самая мощная версия – Data center – предназначена для крупных корпораций.

Windows ME (Millennium Edition).

В Windows ME основное нововведение – новая версия MS Explorer 5,5, обновленный пакет драйверов DirectX 7.1 и несколько новых дополнительных программ (пакет для редактирования видео Movie Maker или универсальный проигрыватель Windows Media Player). Кроме того в систему введена поддержка модных цифровых устройств ввода (цифровых фото и видео номер, усовершенствованная поддержка сканеров).

Полный комплект **Windows ME** занимает на диске от 300 до 500 Мбайт – примерно втрое больше, чем **Windows 98**, соответственно скорость работы снизилась. Для комфортной работы в **Windows ME** необходимо ≥ 96 Мбайт оперативной памяти.

Серьезные изменения потерпела и система безопасности самой ОС, важные изменения произошли и в структуре интерфейса. Среди потерь **Windows ME** следует констатировать потерю части сетевых функций. Поэтому, для больших “корпоративных” сетей теперь рекомендуется **Windows 2000.**

Windows XP.

Изменения в структуре и интерфейсе самой ОС очевидны. Улучшилась защита системных файлов и введен ряд новых драйверов устройств. Одно из серьезных нововведений – встроенная система распознавания голосовых команд и голосового ввода данных. Помимо привычного 32 – разрядного варианта подготовлена 64 – разрядная модификация, предназначенная для установки их сервера с 64 – разрядными процессорами. Windows XP – первая ОС Microsoft с полностью настраиваемым интерфейсом, что очень удобно для пользователей при настройке внешнего вида

Рабочего стола, названия папок , файлов и иконок. Изменения видны в настройке меню «Пуск» и Панели управления. Наиболее приятное нововведение – поддержка записи CD-R и CD-RW дисков на уровне самой ОС. И наконец, в составе Windows XP имеется множество новых и обновленных программ, массу мультимедийных добавлений. Расплата за эти – дополнительные ресурсы компьютера. Для нормальной работы Windows XP требуется не меньше 128 Мбайт оперативной памяти, процессоре частотой не менее 700-800 МТ и около 1 Гбайт дискового пространства.

Заключении можно предполагать, что направление развитие ОС будут следующие:

1. усложнение ОС, при этом дружественный интерфейс;
2. развитие объекта – ориентированной технологии создания ОС, позволяющими компьютеру манипулировать различными объектами;
3. ОС всегда отражают архитектурные решения аппаратной части компьютера. Основная ставка повышение разработки ОС.
4. ОС семейства Windows будет развиваться в направлениях как система для приложений в малом офисе и домашнего использования и как система коллективного использовать для работы в сетях.

Основные термины

Windows NT, Windows 95, Windows 3.1, Windows 98/98 SE, Windows 2000, Windows ME (Millennium Edition), Windows XP.

К о н т р о л ь н ы е в о п р о с ы

1. Что собой представляют операционные системы Windows NT и Windows 95?
2. Расскажите о ОС Windows ME (Millennium Edition).
3. Характеристика о ОС Windows XP.
4. Характеристика о ОС Windows 2000.

Тема 9. Операционные системы новых технологий.

План:

1. Общая характеристика.
2. Архитектура Windows NT.
   a. Модульная структура.
   b. Графический пользовательский интерфейс.
   c. Файловая система.
   d. Характеристика составляющих Windows NT.

ОПЕРАЦИОННЫЕ СИСТЕМЫ НОВЫХ ТЕХНОЛОГИЙ
ОБЩАЯ ХАРАКТЕРИСТИКА

Операционная система (ОС) Microsoft Windows NT - быстродействующая 32-разрядная сетевая операционная система с графическим интерфейсом, встроенными сетевыми средствами и ориентированная на работу в сети.

ОС Windows NT может быть инсталлирована на компьютеры, работающие на платформе Intel 486, Pentium, DEC Alpha, Power PC и MIPS R400.

Для работы Windows NT требуется минимум 16 Мбайт оперативной памяти, для работы в малых сетях необходимо 32 Мбайта, а в более крупных сетях - 64 Мбайт и более.

Минимальный объем жесткого диска сервера должен быть не менее 1 Гбайт и для каждого пользователя еще 100 Мбайт. Каждый сервер должен быть оснащен устройством резервного копирования, а также CD-ROM.

Windows NT поддерживает до 4 Гбайт физической памяти и до 16 Эбайт (экзабайт) дискового пространства (1 Эбайт = 1024 Тбайт = 1 048 596 Гбайт), что способствует использованию RAID-массивов.

Для обеспечения связи между удаленными объектами с помощью сервиса удаленного доступа необходимо наличие модемов на обоих концах соединения. Кроме того, необходимы принтеры, накопители на магнитных лентах (стриммеры) и другие устройства.

В Windows NT реализованы следующие архитектурные решения: переносимость, многозадачность, многопроцессорность, масштабируемость, архитектура клиент-сервер, объектная архитектура, расширяемость, надежность и отказоустойчивость, совместимость, доменная архитектура сетей, многоуровневая система безопасности и др.

Под *переносимостью* понимается способность Windows NT работать на CISC- и RISC-процессорах.

Многозадачность - использование одного процессора для работы множества приложений или потоков нитей (если приложения разбиваются на отдельные исполняемые компоненты).

Многопроцессорная обработка предполагает наличие нескольких процессоров, которые могут одновременно выполнять множество нитей, по одной на каждый имеющийся в компьютере процессор.

Масштабируемость - возможность автоматического использования преимуществ добавленных процессоров. Так, для ускорения работы приложения операционная система может автоматически подключать дополнительные одинаковые процессоры.

Масштабируемость Windows NT обеспечивается:

- многопроцессорностью локальных компьютеров, т.е. наличием у них нескольких процессоров (до 32). Взаимодействие между процессорами осуществляется через разделяемую память;
- симметричной многопроцессорной обработкой, предполагающей возможность одновременного выполнения приложений на нескольких процессорах;
- распределенной обработкой информации между несколькими объединенными в сеть компьютерами. Она реализована на основе концепции вызова удаленных процедур, поддерживающей
  архитектуру клиент-сервер.

Архитектура клиент-сервер предполагает присоединение однопользовательской рабочей станции общего назначения (клиента) к многопользовательскому серверу общего назначения для распределения между ними нагрузки по обработке данных. Их взаимодействие друг с другом имеет объектную ориентацию. Объект, посылающий сообщение, - клиент, а объект, принимающий сообщение и отвечающий на него, - сервер. Объекты могут меняться местами.

Объектная архитектура нашла широкое применение в Windows NT. Объектами являются объекты каталога, объекты процесса и нитей управления, объекты раздела и сегмента памяти, объекты порта и т.д.

Тип объекта включает определенный системой тип данных, набор атрибутов и список операций, которые могут выполняться над ним. Управление объектами могут производить процессы операционной системы. (Процесс - некоторая последовательность действий, определяемых соответствующей программой и составляющих задачу.)

В Windows NT поддерживается распределенная модель объектных компонентов (Distributed Component Object Model - DCOM). DCOM представляет собой систему программных объектов, разработанных для неоднократного использования и замены. Она позволяет разработчикам программного обеспечения создавать составные приложения. DCOM базируется на технологии вызова удаленных программ, что обеспечивает использование механизмов интегрирования распределенных приложений в сети.

Расширяемость Windows NT обеспечена открытой модульной архитектурой, позволяющей добавлять новые модули на все уровни операционной системы. Модульная архитектура обеспечивает возможность соединения с другими сетевыми продуктами. Компьютеры, работающие под управлением Windows NT, могут взаимодействовать с серверами и клиентами других операционных систем.

Характеристики - *надежность* и *отказоустойчивость* - указывают на то, что архитектура защищает операционную систему и приложения от разрушения.

Совместимость означает, что Windows NT версии 4 продолжает поддерживать приложения MS DOS, Windows 3.x, OS/2, а также широкий набор устройств и сетей.

Доменная архитектура сетей предполагает группировку компьютеров в домены.

Для обеспечения безопасности операционной системы, приложений, информации от разрушения, несанкционированного доступа, неквалифицированных действий пользователя в Windows NT разработана *многоуровневая система безопасности* — на уровне пользователя, локальных и сетевых компьютеров, доменов, объектов, ресурсов, сетевой передачи информации, приложений и т.д.

Windows NT сертифицирована на соответствие уровню безопасности C2, являющегося стандартом Департамента безопасности США при работе с конфиденциальной информацией.

Windows NT соответствует следующим требованиям:

- наличию у каждого пользователя уникального имени (идентификатора) и пароля, которые обеспечивают возможность входа в систему и доступа к ее ресурсам;
- возможности управления доступом к ресурсу владельцем ресурса;
- определению различных прав на доступ (особенно на доступ к защищенному объекту);
- организации защищенного канала связи при правильной идентификации компьютеров клиента и сервера), работающих под управлением Windows NT;
- защите системы и ее ресурсов от несанкционированного доступа и несанкционированных изменений (так, для доступа к чужим ресурсам необходимо разрешение пользователя - владельца
  ресурса);
- регистрации всех видов или попыток доступа к защищенной информации или ресурсам компьютера в журнале, доступ к которому ограничен, и т.д.

Однако этот уровень не подразумевает и не гарантирует защиту информации, передаваемой по сети и хранящейся на диске при его переносе на другой компьютер.

Для защиты информации, передаваемой по сети, используются различные методы кодирования и имеется встроенный интерфейс криптографирования - Microsoft Cryptographic Application Program Interface (CryptoAPI).

Интерфейс криптографирования обеспечивает приложениям возможность создания, настраивания и обмена криптографическими ключами, выполнения шифрования/дешифрования и кэширования данных, подключения к системам криптозащиты различных производителей, выборки их по имени либо в соответствии с требованиями системы.

Приложения, в свою очередь, изолированы друг от друга и от аппаратуры, что исключает влияние некорректно работающих приложений на другие и на систему в целом.

Для обеспечения безопасности действий пользователя в домене используется несколько видов контроля: контроль использования пароля пользователями; контроль типов событий, записываемых в журнал безопасности; контроль доверительных отношений доверяемого и доверяющего домена. Кроме того, осуществляется контроль прав доступа пользователя, так как они реализованы на уровне домена и влияют на общую безопасность домена.

Защита от внешних угроз, возникающих при подключении к Интернету, включает: регулярную проверку файлов регистрации, отключение гостевой учетной записи, уничтожение временных файлов, кодирование информации и использование других средств контроля.

АРХИТЕКТУРА WINDOWS NT

МОДУЛЬНАЯ СТРУКТУРА

Windows NT имеет модульную архитектуру (рис. 1). Выделяют два крупных модуля (уровня), каждый из которых состоит из более мелких модулей.

Рис. 1. Модульная структура Windows NT.

Первый уровень - *режим пользователя* (user mode) предоставляет возможность пользователю вступать во взаимодействие с системой. На первом уровне расположены подсистемы среды и подсистема безопасности.

Подсистемы среды - это некоторый набор инструментальных подсистем, поддерживающих разнотипные пользовательские программы. К ним относятся подсистемы: Win-32, поддерживающая 16- и 32-разрядные приложения Windows, приложения DOS и управляющая пользовательским интерфейсом Windows NT; OS/2, поддерживающая приложения OS/2.1.x.

Подсистема безопасности отвечает за легальный вход пользователя в систему.

Второй уровень - *режим ядра* (kernel mode) обеспечивает безопасное выполнение приложений (программ) пользователя. На втором уровне выделяют три укрупненных модуля: исполняющие службы, ядро, уровень аппаратных абстракций.

Исполняющие службы отвечают за взаимодействие между ядром подсистемы и подсистемами среды приложений. В состав исполняющих служб включены системный сервис и службы режима ядра.

Системный сервис является интерфейсом между подсистемами среды приложений и службами режима ядра.

К *службам режима ядра* относятся следующие программные модули:

- *диспетчер ввода-вывода*, обеспечивающий управление процессами ввода-вывода информации;
- *диспетчер объектов*, управляющий системными операциями, производимыми над объектами, такими, как использование, переименование, удаление, обеспечение защиты объекта;
- *диспетчер контроля безопасности*, обеспечивающий модель безопасности системы;
- *средства вызова локальных процедур*, поддерживающие работу пользовательских приложений и подсистем среды и обеспечивающие обмен информацией;
- *диспетчер виртуальной памяти* - служба, управляющая физической и виртуальной памятью;
- *диспетчер процессов*, управляющий действиями процессов: созданием, удалением, протоколированием; распределяющий адресное пространство и другие ресурсы между процессами.

Ядро Windows NT управляет всеми системными процессами, обеспечивает оптимальное функционирование системы.

Уровень аппаратных абстракций - часть системы, обеспечивающая независимость верхних уровней операционной системы от особенностей и различий конкретной аппаратуры. В этом модуле хранится вся аппаратно-зависимая информация.

ГРАФИЧЕСКИЙ ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

Графический пользовательский интерфейс предназначен для создания пользователю комфортных условий при работе с операционной системой Windows NT. Интерфейс Windows NT интуитивно понятный, простой и удобный. Он удобен при запуске программ, открытии и сохранении файлов, работе с файлами, дисками и сетевыми серверами. Графический многооконный пользовательский интерфейс GUI (Graphics User Interface) в Windows NT основан на реализации объектно-ориентированного подхода, при котором работа пользователя ориентирована в первую очередь на документы, а не на программы. Загрузку любого имеющегося документа можно осуществить путем открытия файла, содержащего этот документ, одновременно автоматически загрузится программа, с помощью которой открываемый файл был создан.

Пользовательский интерфейс Windows NT включает следующие элементы: Рабочий стол; Панель задач; Стартовое меню; Контекстное меню; Систему меню приложений Windows; ярлыки: Мой компьютер, Сетевое окружение, Корзина, Проводник Интернета, Входящие, Портфель; Окно; Шрифты; Справочная система Windows NT.

В пользовательском интерфейсе Windows NT заложена концепция *ярлыков*. Почти все, что расположено на Рабочем столе Windows NT, - это ярлыки. Ярлыки представляют собой маленькие файлы, связанные с соответствующими объектами. Они могут храниться в любой *Папке,* включая *Рабочий стол*. Ярлык - это не сам объект, а указатель на него. Это значит, что можно создавать и удалять ярлыки, и это не будет влиять на сами объекты.

Ярлыки обеспечивают быстрый доступ к таким объектам, как программы, папки, документы, устройства компьютера или сети. Для этого следует два раза щелкнуть на соответствующем ярлыке.

Ярлыки представляются в виде специального значка (пиктограммы). В создаваемых пользователем ярлыках в отличие от системных в левом нижнем углу располагается черная стрелка.

Внешний облик Windows NT, отображаемый на экране монитора, представлен в виде *Рабочего стола*, все элементы которого - ярлыки, изображающие программы, документы, устройства, - расположены в удобном для пользователя порядке (рис. 2).

Пользователь может оформить свой *Рабочий стол* и расположенные на нем элементы в соответствии со своим художественным вкусом - изменить расположение и внешний вид ярлыка, форму и цвета отдельных элементов, оформление экрана. Пользователь может расположить на экране одновременно окна, содержащие фрагменты нескольких приложений, необходимых в данный момент в работе, аналогично разложенным папкам с документами на *Рабочем столе*.

Рис. 2. Поверхность **Рабочего стола.**

Все элементы, расположенные на *Рабочем столе* Windows NT 4, являются объектами, обладающими определенными свойствами, и ими можно манипулировать.

Рабочий стол является хранилищем, папкой, в которой могут содержаться компьютеры, диски, файлы и другие папки.

Папка - место для хранения программ, документов и дополнительных папок. Папка в Windows NT представляет аналог каталога, директории в MS DOS.

Рабочий стол - это тоже папка, находящаяся на вершине иерархии папок Windows NT. Но эта папка не закрывается и не открывается. Она всегда присутствует на экране.

В нижней части Рабочего стола расположена *Панель задач,* являющаяся основным средством взаимодействия пользователя с системой.

Слева, на *Панели задач,* находится кнопка **Пуск,** за которой следуют кнопки с именами открытых приложений, а справа индицируются текущая раскладка клавиатуры, время и др. (рис. 3).

Рис. 3. Панель задач

При нажатии кнопки **Пуск** открывается *Стартовое меню*. Активизация кнопок с именами работающих приложений позволяет быстро переключаться в нужное приложение. При необходимости *Панель задач* можно переместить в другое место экрана или сделать ее исчезающей, если щелкнуть кнопкой мыши на пустом месте *Панели задач* и, удерживая ее в нажатом состоянии, перетащить на новое место.

Стартовое меню открывается в левой нижней части *Рабочего стола* непосредственно над кнопкой **Пуск.** Стартовое меню обеспечивает пользователю доступ почти ко всем функциям Windows NT: открытие документа; запуск приложения; быстрый поиск документов, находящихся на локальных и сетевых дисках (по имени, типу, размеру, дате, содержанию); выполнение настройки компьютера; получение справочной информации; завершение работы в Windows NT.

Стартовое меню состоит из нескольких пунктов. Справа от некоторых элементов стартового меню отображается черный треугольник, показывающий, что этот элемент тоже является меню (папкой).

Каждая из помеченных треугольником папка имеет свое каскадное подменю, которое раскрывается сразу же при установке курсора указателя на соответствующем пункте. При необходимости можно добавлять или удалять пункты *Стартового меню*.

Иногда имена наиболее часто используемых приложений или документов размещаются в верхней части *Стартового меню* над стандартными пунктами.

Контекстное (контекстно-зависимое) меню вызывается щелчком правой кнопки мыши. Оно содержит некоторый список свойств, или набор команд (или то и другое), доступных в данный момент для работы с выбранным объектом. Список команд зависит от конкретного объекта. С помощью этого меню осуществляется быстрый доступ к свойствам объектов, нужным командам.

Контекстные меню широко используются в Windows NT. Они доступны в любом месте интерфейса.

В левой части *Рабочего стола* расположена группа ярлыков с названиями, соответствующими их бытовым аналогам.

Мой компьютер вызывает соответствующую универсальную программу, обеспечивающую быстрый доступ ко всем элементам системы - локальным и сетевым дискам, принтерам, контрольной панели и т.д. Активизация ярлыка вызывает открытие окна с ярлыками, соответствующими сетевым ресурсам.

Сетевое окружение - папка, содержащая ярлыки всех компонентов рабочей группы или домена, а также ярлык *Вся сеть*, предоставляющий доступ к другим доменам и рабочим группам.

Входящие - ярлык папки входящей корреспонденции появляется на *Рабочем столе* лишь при поддержке электронной почты, обеспечивает реализацию средств управления входящими и выходящими документами, являющимися объектами функционирования электронной почты или факса.

Корзина - ограниченная область памяти на жестком диске (может быть организована для каждого диска) компьютера, служит местом хранения удаленных файлов. Корзина запоминает имя, исходное местоположение, дату удаления, тип и размер всех удаленных файлов. Удаленные файлы могут быть восстановлены. Активизация корзины вызывает открытие окна папки со списком последних удаленных файлов. При переполнении корзины файлы, хранящиеся в корзине дольше всех, удаляются безвозвратно.

Портфель - специальная папка для пользователей, работающих с настольными и портативными компьютерами, - область памяти, в которую помещены документы, с которыми осуществляется работа в разных местах: дома, на работе и т.д. Файлы, копируемые в портфель и из него, автоматически обновляются на портативных компьютерах (т.е. последняя версия заменяет предыдущую).

Проводник Интернета - Web-браузер от Internet - вызывает работу программы *Проводник Интернета* (Internet Explorer) для просмотра всех доступных данных и отправки новых сообщений и вызывает ее на экран.

Одним из основных понятий пользовательского интерфейса Windows NT является *окно*, как ограниченная прямоугольной рамкой поверхность экрана. В окне Windows NT отображаются папки и файлы, выполняемые программы и документы.

Окно представляет собой прямоугольник, размер которого может изменяться пользователем. Окно может быть нормальным (2/3 экрана), полноэкранным, произвольным и свернутым, представленным в виде кнопки с подписью (в свернутом окне программа продолжает выполняться).

С помощью манипулятора мышь можно перемещать окна по экрану, менять их размер, цветовую гамму окон и составляющих элементов, раскрывать и закрывать их.

Некоторые приложения используют окна, разделенные на части в горизонтальном и вертикальном или в обоих направлениях. Эти части называются областями. С помощью разделителя можно изменять их относительные размеры.

Все элементы оконного интерфейса стандартизированы: ниже верхней границы окна расположен выделенный цветом заголовок с именем папки или выполняемой программы, еще ниже - строка меню, а затем - рабочее поле (содержимое окна).

Существуют два основных типа окон - окна приложений и окна документов, а в случае необходимости внесения разъяснений для пользователя системой открываются диалоговые окна.

Окно приложения содержит программу или папку. Окно программы содержит в верхней части заголовок, состоящий из эмблемы и имени программы, имени документа, а также кнопки управления окном. Ниже расположена строка меню. Центральную часть окна занимает в зависимости от активного приложения рисунок, текст, таблица и т.д.

Рядом с правой и нижней границами окна расположены полосы прокрутки, позволяющие осуществлять вертикальное и горизонтальное перемещения по обрабатываемому документу.

Окно папки отражает содержимое папки. Оно напоминает окно небольшого приложения и имеет заголовок, кнопки для изменения размеров и закрытия окна, значок системного меню, строку собственного меню, дополнительную панель инструментов и строку состояния. Заголовок окна папки по умолчанию содержит название, указанное под значком данной папки (рис. 4).

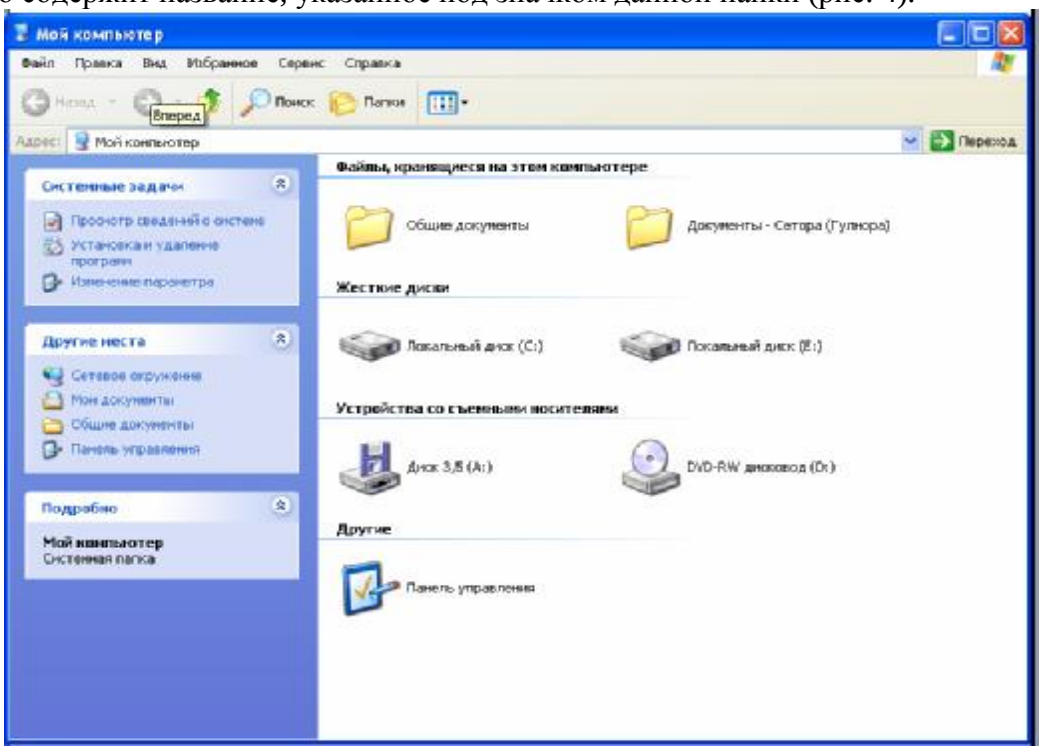


Рис. 4. Окно папки.

Системное меню приложения обеспечивает доступ к базовому набору команд, общих для всех приложений. Строка меню представляет собой горизонтальную полоску, расположенную непосредственно под заголовком, с командами, присущими конкретному приложению. Каждая команда открывает свое меню.

Окно папки может быть просмотрено в одном из четырех режимов: Крупные значки, Мелкие значки, Списки, Таблица.

Окно документа расположено внутри окна приложения и всегда остается в пределах окна своего приложения. В верхней части окна документа расположен заголовок, включающий эмблему, имя документа, а также кнопки управления окном. В центральной части - содержание документа, внизу и справа - полосы прокрутки. Свернутые окна документа преобразуются в значок с миниатюрным заголовком.

Окно диалога содержит строку заголовка, кнопки управления окном, а также может включать несколько вкладок и содержать следующие элементы: командную кнопку, переключатели, текстовое поле информации, флажки, раскрывающиеся списки и др.

В пользовательском интерфейсе Windows NT реализован принцип WYSIWYG: то, что вы видите на экране, будет перенесено на бумагу независимо от типа устройства вывода.

Такая возможность появляется в связи с тем, что в Windows встроена поддержка контурных шрифтов формата True Type, не зависящих от типа принтера.

Интерфейс администратора сети не отличается от интерфейса пользователя. Графический интерфейс значительно облегчает работу администратора сети Windows NT. Так, информация о новых пользователях вводится администратором в графические формы, отображаемые на экране. Определение организационных групп и предоставление прав доступа осуществляются несколькими операциями с помощью технологии *буксировка и освобождение* (Drag and drop). Изменение группы, в которую помещается учетная запись пользователя, и соответственно автоматическое изменение его прав осуществляются путем перетаскивания идентификатора пользователя из одной группы в другую и т.д.

ФАЙЛОВАЯ СИСТЕМА

Файловая система является важнейшим компонентом Windows NT. Windows NT поддерживает разные типы файловых систем.

Для работы с разными типами файловых систем в Windows NT построена аппаратно-независимая модель подсистемы ввода-вывода. Она реализована на концепции многоуровневой архитектуры драйверов и устройств в сочетании с диспетчером ввода-вывода, который является посредником между прикладными программами и драйверами.

ОС Windows NT поддерживает следующие файловые системы:

FAT (File Allocation Table) - стандарт для MS DOS;

NTFS (New Technology File System), разработанную специально для Windows NT;

CDFS (Compact Disc File System) - специальную файловую систему для CD-ROM-накопителей.

Кроме того, Windows NT Server поддерживает в NTFS-разделах файловую систему Macintosh.

FAT используется, когда необходима совместимость с такими операционными системами, как MS DOS, Windows 95 и др. Рекомендуется FAT-разделы преобразовывать в NTFS-разделы, при этом потери информации не происходит. Обратное преобразование информации не рекомендуется, так как часть информации теряется.

NTFS - основная файловая система Windows NT, ее разделы доступны только из Windows NT, и поскольку на сервере должна быть установлена только одна файловая система, то рекомендуется устанавливать NTFS.

NTFS превосходит FAT по скорости работы и по эффективности использования ресурсов. Она предназначена для построения компьютерных систем от рабочей станции до сервера предприятия класса мэйнфреймов и может работать с дисковыми томами, содержащими до 2^{64} байт информации.

В NTFS реализована эффективная методика сжатия данных и динамического кэширования диска. Сжатие осуществляется параллельно с чтением следующего блока данных, а при передаче в кэш-память данные декомпрессуются. Уменьшение размера большинства текстовых файлов - 50%, исполняемых - 40%, степень сжатия баз данных - еще выше.

Windows NT поддерживает виртуальный режим работы файловой системы.

NTFS - это сложная реляционная база данных, применяющая новейшие технические достижения для протоколирования и восстановления данных.

Система NTFS - самовосстанавливающаяся, т.е. при любых сбоях можно своевременно восстановить данные и возобновить работу системы. Это обеспечивается наличием записи, соответствующей каждому файлу, в таблице MFT (Master File Table); избыточностью данных; протоколированием в специальном файле всех операций с данными (запись; удаление; переименование; изменение данных, атрибутов и индексов и т.д.), выполняемых на диске NTFS, и другими средствами.

NTFS обеспечивает безопасное хранение и передачу данных. Для хранения данных организовано отказоустойчивое дисковое хранилище, основанное на зеркализации данных (дублировании) на дополнительном диске, на использовании технологии создания больших дисковых пространств - RAID-массивов (ряда независимых дисков, образующих единый логический диск). Кроме того, NTFS обеспечивает защиту сменных дисков, отформатированных для NTFS, теми же механизмами контроля, что и постоянные диски.

NTFS позволяет устанавливать права доступа даже для отдельных файлов. В этой системе реализован объектно-ориентированный подход к управлению файлами, поэтому каждый файл или каталог рассматривается как объект, описываемый атрибутами, которые задаются пользователем или системой.

Атрибутами являются все признаки, идентифицирующие файл (например, имя файла, дескриптор защиты, сами данные, время последней модификации, счетчик связей, сведения о носителе информации и др.).

Файловая система предоставляет возможность работы с двумя вариантами имен файлов - длинным и коротким.

Длинное имя файла (именуемое Primary Filename, т.е. первичное) может иметь длину в пределах 255 символов, содержать символы UNICODE, несколько точек и пробелы внутри имени. При необходимости длинное имя преобразуется в короткое.

Короткое имя файла (Alias - псевдоним) является именем - заменителем и имеет формат "8+3". Alias-имя генерируется ОС автоматически при необходимости доступа к файлам с длинными именами из ранних версий Windows- и DOS-приложений. При этом все буквы преобразуются в прописные, из имени исключаются специальные символы и, кроме того, длинное имя укорачивается.

Для того чтобы обратиться к какому-либо файлу, следует указать место его хранения - его путь. Путь начинается с имени диска, на котором записан файл, затем ставятся двоеточие **":",** обратная косая черта "\" и далее перечисляется последовательность всех имен папок, которые необходимо открыть, чтобы получить доступ к файлу, включая имя файла.

Чтобы указать путь к файлу, находящемуся в сетевой папке, надо ввести после имени диска две обратные косые черты.

Группа файлов на одном магнитном диске может быть объединена по какому-либо критерию в папку. Папки и файлы, в свою очередь, могут быть объединены в папку более высокого уровня (родительскую). Уровень вложенности папок не ограничивается.

Windows NT имеет сложную иерархическую структуру файлов и папок. На верхнем уровне иерархии расположен *Рабочий стол*. Следующий уровень представлен папками, расположенными в левом верхнем углу *Рабочего стола*, и значками с подписями: *Мой компьютер*, *Сетевое окружение*, *Корзина*, *Портфель* и т.д.

Если открыть папку *Мой компьютер*, то далее по нисходящей расположены уровень дисководов, логических дисков, устройство чтения компакт-дисков, панель управления, принтеры, удаленный доступ к сети. Еще ниже расположены папки приложений, файлы документов и программ.

При открытой папке *Сетевое окружение* по нисходящей расположен значок *Вся сеть*, на следующем уровне - значки, указывающие, на основе каких сетей построена вся сеть, далее - значки доменов и рабочих групп, входящих в сеть Windows NT, далее -имена или номера компьютеров, входящих в конкретные рабочие группы, и значки устройств, подключенных к конкретному компьютеру.

Структуру папок на диске можно просмотреть с помощью программы *Проводник Windows NT*.

ХАРАКТЕРИСТИКА СОСТАВЛЯЮЩИХ WINDOWS NT

Основой Windows NT является ОС Windows NT Server, предназначенная для управления сетевыми ресурсами. Другой частью системы Windows NT является ОС Windows NT Workstation, обеспечивающая рабочее место клиента.

В настоящее время широкое распространение получили версии Windows NT Server 4 и Windows NT Workstation 4.

Windows NT Server 4 - мощная многоцелевая операционная система, которая может осуществлять организацию централизованного хранения большого количества коллективно используемых файлов, выполняя при этом функции сервера файлов и функции сервера приложений, таких, как СУБД, финансовые системы и др.; позволяет подключать и предоставлять в совместное пользование множество локальных или сетевых принтеров; осуществлять резервное копирование данных на стриммер.

Windows NT Server 4 может выполнять функцию сервера службы удаленного доступа - связь клиента с ресурсами сети, обеспечивая доступ к файлам, печать документов, подключение к хостам, обмен сообщениями по электронной почте и т.д. Функции клиента могут выполнять компьютеры с установленными на них операционными системами MS DOS, Windows for Workgroups, Windows 95, Windows NT Workstation.

Количество одновременных сеансов, организуемых службой удаленного доступа, - 256, а количество одновременно устанавливаемых соединений для параллельного использования ресурсов сети - неограничено.

В Windows NT Server 4 реализован протокол многоканальной связи PPTP (Point to Point Tunnelling Protocol). Он позволяет организовать виртуальные корпоративные сети путем соединения локальных сетей через Internet. Для построения глобальной сети в систему Windows NT Server 4 встроена возможность многопротокольной маршрутизации, что позволяет использовать Windows NT Server 4 в качестве маршрутизатора между различными локальными и глобальными сетями с различной топологией.

Windows NT Server 4 может быть использована для построения как простейшей сети из нескольких персональных компьютеров, так и для сложной гетерогенной системы на сотни тысяч пользователей, в которых компьютеры группируются в *домены*. Она поддерживает наиболее распространенные в настоящее время сетевые протоколы: NetBEUI (совместимый с IPX/SPX), TCP/IP, AppleTalk и т.д., работает с большинством клиентских систем, с системами на базе Novell NetWare, Unix и др.

Windows NT Workstation 4 - менее мощная ОС по сравнению с Windows NT Server 4 разрабатывалась как рабочая станция (или расширенная настольная система). Эта ОС может быть использована как невыделенный сервер в одноранговых сетях, а также в качестве клиента сетей на базе Windows NT Server 4, к которым предъявляют повышенные требования по обеспечению надежности, защите информации, соблюдению условий разграничения прав доступа, а также в сетевых серверных системах. На основе Windows NT Workstation создаются *рабочие группы*.

Windows NT Workstation может выполнять функции файл-сервера, если пользователям выделен каталог с помощью Windows NT Explorer для совместного использования. Она позволяет совместно использовать принтеры и другие устройства участникам рабочей группы, в состав которой входит конкретный компьютер. Однако функции сервера в Windows NT Workstation ограничены малыми сетями (параллельное использование разделяемых ресурсов позволено не более 10 пользователям), а также ограниченной возможностью организации защиты информации.

Windows NT Workstation 4 предоставляет пользователю стандартные средства для работы отдельных пользователей и групп пользователей. Кроме того, имеются специфические средства динамического компрессирования файлов, управления правами доступа к отдельным каталогам и файлам, контроля исполнения приложений, системных служб, просмотра динамического графика использования памяти и процессора, резервного копирования на магнитную ленту.

Windows NT Workstation 4 предлагает пользователю средства навигации по серверам в Интернете, системы обмена сообщениями, инструмент создания Web-страниц и распространения информации внутри корпоративных Интернет-сетей, средства актуализации базы данных сервера, системы имен доменов, службы адресации в Интернете по IP-адресам хост-компьютеров. Кроме того, Windows NT Workstation 4 поддерживают технологии телефонной связи, факсимильной связи, системы электронной почты.

В Windows NT Workstation 4 включены: графический редактор; программы звукозаписи, текстовый редактор; интерфейсы программирования приложений.

Windows NT Workstation 4 допускает подключение одного удаленного пользователя. Она содержит средства подключения к серверу удаленного доступа в качестве клиента, а также средства, обеспечивающие вывод информации (в том числе графической) с Windows NT Workstation 4 на сервер печати Windows NT Server 4.

В Windows NT Workstation 4 предусмотрена возможность интеграции с сетями на платформе Novell NetWare 4.x, которая реализуется на основе встроенной в Windows NT Workstation 4 глобальной службы каталогов Novell - NetWare Directory services. Это обеспечивает вход в систему NetWare, доступ к файлам и каталогам.

Основные термины

Многозадачность, переносимость, многопроцессорная обработка, масштабируемость, архитектура клиент-сервер, объектная архитектура, расширяемость, доменная архитектура сетей, системный сервис, графический пользовательский интерфейс, корзина, сетевое окружение, мой компьютер, портфель, проводник Интернета, окно приложения, окно документа, файловая система, атрибут.

К о н т р о л ь н ы е в о п р о с ы

1. Что собой представляют операционные системы Windows NT?
2. Какие отличательные особенности сетевой операционной системы MS Windows NT вы знаете?
3. В чём сущность и назначение доменов в доменной архитектуре сети?
4. Чем обеспечивается безопасность данных ОС Windows NT?
5. Какова назначения ярлыков в интерфейсе ОС Windows NT?
6. Расскажите о видах и операциях манипулирования с окнами.
7. Какие типы файловых систем поддерживаются ОС Windows NT и в чём их особенности?
8. Приведите составляющие ОС Windows NT и дайте им краткую характеристику?

Тема 10. Прикладные программные средства офисного назначения. (4ч)

План:

1. Общие сведения.
2. Принципы работы программных продуктов семейства MICROSOFT OFFICE.
3. Текстовые редакторы.
4. Табличные процессоры.

ПРИКЛАДНЫЕ ПРОГРАММНЫЕ СРЕДСТВА ОФИСНОГО НАЗНАЧЕНИЯ

ОБЩИЕ СВЕДЕНИЯ

В настоящее время на рынке программного обеспечения имеются мощные программные пакеты, получившие название офисных систем. К наиболее популярным офисным системам следует отнести Microsoft Office фирмы Microsoft и Lotus Notes фирмы Lotus Development. Каждый из офисных пакетов содержит текстовый редактор, электронные таблицы, средства для создания и поддержки баз данных, средства коммуникаций.

Наиболее распространенным в Узбекистане и в настоящее время является пакет Microsoft Office. Это связано с тем, что фирма Microsoft - автор Windows и Microsoft Office (MS Office) - логично вписывается в интерфейс Windows. Понимая логику работы с Windows, достаточно легко освоить

прикладные окна программных средств, входящих в MS Office. Кроме того, совместное выполнение ряда программных средств, входящих в MS Office, позволяет гибко распределять их ресурсы и работу, увеличивать общую производительность. Надо отметить, что другие производители программных продуктов для электронных офисов готовят специальные версии для работы под Windows. Однако для создания интерфейса между MS Office и другими офисными пакетами, например Lotes Notes, необходимо наличие специального программного обеспечения. Для сохранения, упорядочения и распространения документов из MS Office в Lotes Notes разработан специальный пакет OfficeLink for Lotus Notes.

В MS Office входят текстовый редактор Word, табличный процессор Excel, средство для создания баз данных Access, а также специальные программы для организации работы офисов. Среди этих программ Microsoft Outlook - средство доступа к разнообразной информации и ее коллективной обработки, PowerPoint -мощное приложение для подготовки и проведения презентаций, FrontPage - приложение для создания Web-страниц и ряд других.

В связи с тем, что система Windows постоянно модифицируется фирмой-производителем, версии программного пакета MS Office соответственно изменяются вместе с ней. Для Windows 95 разработан MS Office 95, который включает Word 7.0, Excel 7.0 ит.д.

В 1997 г. появились новая версия офисного пакета - MS Office 97 и входящие в него программные средства Word 97, Excel 97, Outlook 97 и т. п. Эти версии содержат много новаций по сравнению с предыдущими, особенно в части улучшения связи с информационной сетью Интернет.

В связи с появлением версии операционной системы Windows 2000 разработана новая версия офисного пакета MS Office 2000. Основная особенность перечисленных операционных систем - высокий уровень интеграции с Интернетом. Пользовательский интерфейс подчинен этой цели: локальные диски выглядят также, как узлы Web. При этом возможен выбор между классическим и Web-ориентированным интерфейсом. В настоящее время начал внедряться пакет Microsoft Office 2000. В данной версии проводится дальнейшая интеграция с Интернетом. HTML применяется в качестве полноценного формата файлов.

В описаниях данной версии отмечается, что она выходит за границы традиционных настольных систем, превратившись в корпоративное приложение для предприятий любого масштаба. Данную версию офиса можно рассматривать как платформу для создания специализированных решений или клиентское средство доступа к корпоративным данным.

В целом основная тенденция в развитии программных продуктов данного пакета - повышение "интеллектуальности".

Сюда можно отнести улучшенный инструментарий для коррекции грамматических ошибок. С каждой новой версией офисных продуктов улучшаются встроенные в эти средства возможности грамматического и лексического контроля. В новой системе меню предполагается отображать только наиболее часто используемые функции, причем каждый пользователь может настраивать структуру меню "под себя".

На каждого специалиста из офиса любого уровня возлагаются определенные функциональные обязанности, непосредственно влияющие на выбор состава и возможностей используемых программ. При этом следует ориентироваться на доступные для данного пользователя программные средства. Например, для секретаря или документоведа основной упор следует делать на возможности текстового редактора; для выпускника экономического вуза (квалифицированного экономиста, бухгалтера, банкира) знание текстового редактора должно сочетаться со знанием возможностей табличного редактора и умением работать с ним. Большим подспорьем в работе экономиста и финансиста может стать знание основных возможностей СУБД как инструмента формирования расчетных документов.

Для руководителя офиса (главного бухгалтера, финансового директора и т.п.) необходимо иметь представление о новых возможностях программных средств MS Office для совместной работы многих пользователей. Помимо этого большое значение для организации управления офисом имеет возможность информационного обмена между сотрудниками. Данная задача решается с помощью клиентского приложения, входящего в MS Office, -Microsoft Outlook.

ПРИНЦИПЫ РАБОТЫ ПРОГРАММНЫХ ПРОДУКТОВ СЕМЕЙСТВА MICROSOFT OFFICE

Несмотря на разнообразие появившихся версий программных продуктов семейства MS Office, в них заложены единые принципы построения и работы с ними. Это позволяет, овладев одним средством, в дальнейшем достаточно легко освоить его новые версии.

В качестве базовых программных средств, представляющих работу текстового редактора и табличных процессоров, выбраны Word 2003 и Excel 2003, поскольку эти программные средства в настоящее время наиболее распространены. На них будет продемонстрирован тот минимум возможностей, которого достаточно для работы специалиста в финансово-экономической сфере деятельности. В случае необходимости получения дополнительных знаний следует обратиться к специальной литературе.

Текстовый редактор, табличный процессор, базы данных в первую очередь предназначены для обработки данных. Основные операции, которые пользователь производит над данными, заключаются в их вводе, редактировании, копировании, перемещении и удалении.

Некоторые действия во всех пакетах MS Office несут одну и ту же нагрузку и выполняются аналогично. Сюда можно отнести команды (или соответствующие им кнопки на панели инструментов):

из пункта меню Файл:

Создать - создает новую рабочую книгу (документ, базу данных);

Открыть - открывает существующий файл;

Закрыть - закрывает открытую рабочую книгу (документ, базу данных);

Сохранить - сохраняет изменения в текущей рабочей книге (документе, базе данных);

Сохранить как - сохраняет рабочую книгу (документ, базу данных) под новым именем или с новым адресом;

из пункта меню Правка:

Отменить - отменяет выполненное действие;

Повторить - возвращает последнее действие;

Вырезать - вырезает и помещает выделенную область в буфер обмена Windows (Clipboard);

Копировать - копирует выделенную область в буфер обмена Windows;

Вставить - вставляет содержимое буфера обмена Windows в текущую таблицу (документ, базу данных);

из пункта меню Вид:

Полный экран - скрывает все элементы экрана (панели инструментов, меню, полосы прокрутки, строку заголовка, линейку, область стиля и строку состояния);

Панели инструментов - создает, показывает, скрывает панели инструментов.

Работа с панелями инструментов во всех программных продуктах семейства MS Office происходит аналогичным образом. На экране показаны те панели инструментов, которые помечены "галочкой" в меню **Панели инструментов** (рис.1). Панели можно перемещать по экрану и располагать их в удобном месте. Принцип перемещения панелей по экрану полностью идентичен перемещению окон в Windows. Кроме того, имеется возможность создать собственную панель инструментов или внести коррективы в имеющиеся, воспользовавшись кнопкой **Настроить** в окне *Панели инструментов*.

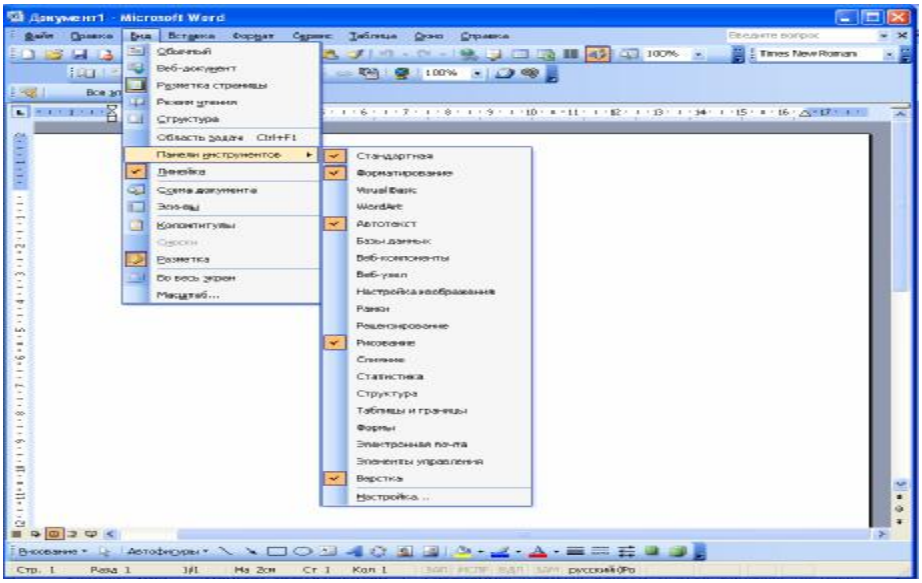


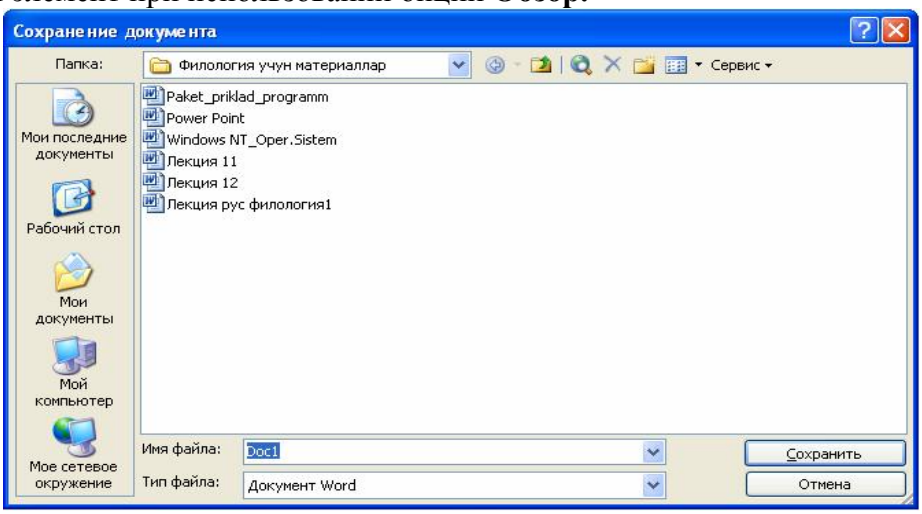
Рис. 1. Окно Панели инструментов

Если в созданный документ надо добавить схему или какой-либо рисованный объект, используется панель инструментов *Рисование*, которая вызывается на экран последовательностью команд: **Вид -> Панели инструментов -> Рисование.** Порядок работы с этой панелью во всех программах MS Office одинаков, а именно:

1. выбрать нужный инструмент на панели *Рисование* и щелкнуть левой кнопкой мыши. Курсор превратится в маленький крестик;
2. подвести курсор к нужному месту на рабочем поле документа;
3. нажать левую кнопку мыши и, не отпуская ее, нарисовать фигуру, соответствующую выбранному инструменту;
4. отпустить левую кнопку мыши.

Нарисованный объект можно отформатировать: изменить размер, переместить на другое место, разъединить объект на группы, сгруппировать несколько объектов, изменить стиль, цвет, толщину линии Кроме того, нарисованный объект можно скопировать, вырезать, вставить, удалить.

Приведенная группа команд из пункта меню *Файл* требует умения работать с диалоговым окном (рис. 2), с которым пользователь встречается уже при работе в среде Windows, например создавая новый программный элемент при использовании опции **Обзор.**



Это связано с существующим специальным соглашением, которое регулирует структуру организации диалоговых окон в этой среде Отличие состоит лишь в информации, помещенной в заголовке окна, которая соответствует выполняемому действию, и в форматах предлагаемых файлов. При работе с данным диалоговым окном пользователь поочередно выполняет следующие действия:

1. выбирает тип файла;
2. выбирает диск, на котором размещен файл;
3. выбирает папку, в которой расположен файл;
4. выбирает или задает имя файла;

5. подтверждает выполненные действия, нажав кнопку **ОК.**

Неотъемлемой частью любого пакета программ является модуль редактирования информации. Во всех пакетах семейства MS Office программы операции редактирования - **Вырезать, Копировать, Вставить** - выполняют одни и те же функции и производятся одинаковым образом.

Копирование данных - это операция, которая позволяет размножать данные из одних мест таблицы (документа, базы данных) в другие. При копировании исходные данные сохраняются Копирование можно производить разными способами.

Для копирования данных с помощью мыши необходимо:

1. выделить нужный диапазон;
2. подвести курсор мыши к границе выделенного диапазона;
3. нажать клавишу [Ctrl] и удерживать ее;
4. не отпуская кнопку мыши, переместить выделенный диапазон в нужное место;
5. отпустить кнопку мыши и клавишу [Ctrl].

Применяя команды редактирования, необходимо:

1. выделить нужный диапазон;
2. выполнить команду **Правка / Копировать;**
3. установить курсор в нужное место;
4. выполнить команду **Правка / Вставить.**

Перемещение данных - это операция, имеющая принципиальное отличие от операций копирования: при перемещении данные удаляются из исходного места, где они первоначально находились. Перемещение данных можно проводить с использованием мыши и буфера обмена Windows.

*Для **перемещения данных с помощью мыши*** необходимо:

1. выделить нужный диапазон;
2. подвести курсор мыши к границе выделенного диапазона;
3. нажав на левую клавишу мыши и не отпуская ее, переместить выделенный диапазон в нужное место;
4. отпустить кнопку мыши.

Применяя команды редактирования, необходимо:

1. выделить нужный диапазон;
2. выполнить команду **Правка / Вырезать;**
3. установить курсор в нужное место;
4. выполнить команду **Правка / Вставить.**

Как видно из приведенных примеров, процедура копирования и перемещения заканчивается выполнением команды **Вставить** Эта команда помещает содержимое буфера обмена в указанное место.

Все приведенные выше операции также можно выполнить следующими способами:

1. применяя команды экспресс-меню;
2. применяя кнопки панели управления **(Вырезать), (Копировать), (Вставить).**

Экспресс-меню, или иначе его называют *контекстное меню*, используется помимо основного меню, постоянно находящегося на экране. При активизации контекстного меню на экране появляются команды, относящиеся к активному объекту. Например, если в данный момент активен рисунок, то на экране появится экспресс-меню работы с ним.

Для активизации экспресс-меню достаточно:

1. выделить нужный диапазон;
2. установить курсор мыши на закрашенную область;
3. нажать *правую* кнопку мыши.

На экране появится окно меню, которое используется аналогично меню **Правка.**

Следующим общим принципом в работе с программами семейства MS Office является возможность использования справки. Чтобы запросить справку, соответствующую ситуации, в которой программа находится в данный момент, необходимо нажать на клавишу **[F1]** или на кнопку вызова помощника, или выполнить пункт меню <? >.

ТЕКСТОВЫЕ РЕДАКТОРЫ

Текстовый редактор - это программное средство для подготовки текстовых документов. Существует много программных средств этого назначения, начиная от самых простых, например редактор WordPad, входящий в состав Windows, до сложных издательских систем. При подготовке на компьютере различных деловых документов, отчетов и т.п. необходимо использовать текстовые редакторы, занимающие промежуточное положение между простейшими редакторами и издательскими системами. Эти редакторы, с одной стороны, достаточно доступны в изучении и не требуют сложной и дорогостоящей техники, с другой - имеют все средства, необходимые для создания сложных документов. Часто в литературе эти программные средства называются *текстовыми процессорами*. Рассмотрим возможности таких редакторов на примере текстового редактора Word, входящего в пакет MS Office.

Редактор Word достаточно прост для освоения и позволяет выполнять многие операции, присущие издательским системам. Освоив работу с этим редактором, пользователь может иметь у себя на столе удобную издательскую систему и избавиться от участия в подготовке своих работ не только машинистки, но и редактора, а то и художника.

Разработан WinWord как приложение операционной системы Windows, и по своему интерфейсу он очень похож на нее.

Для Windows 95/98 и Windows NT в настоящее время наиболее целесообразно применять текстовый редактор из MS Office97. Это связано с тем, что программные средства, входящие в данный пакет, имеют большие возможности для создания файлов и работы с ними в формате HTML.

Надо иметь в виду, что для нормальной работы с этими программными средствами компьютер должен иметь характеристики не хуже, чем:

- процессор Pentium 133/166;
- ОЗУ емкостью 16 Мбайт;
- жесткий диск емкостью 1,2 Гбайт.

Чем лучше характеристики компьютера, тем быстрее и комфортнее работает текстовый редактор.

В настоящее время инсталляция MS Office 97 осуществляется с CD-ROM. В соответствующей папке ищется файл setup.exe. После запуска данного файла необходимо строго придерживаться указаний программы-установки. Причем возможна как установка всего пакета, т.е. Word97, Excel97, Access97 и ряда других программных средств, так и установка некоторых из этих программ.

При установке Office97, как и любого другого программного средства, необходимо решить вопрос: устанавливать ли локализованную версию системы или нет. *Локализованной* называется система, которая реализована на языке той страны, где она устанавливается. Конечно, работать в текстовом редакторе, где все подсказки, меню, сообщения и т.п. выдаются на русском языке, удобнее. Поэтому далее рассматриваются локализованные версии программных продуктов, учитывая, что они достаточно распространены в Узбекистане.

Для запуска Word из Windows необходимо подвести курсор к пиктограмме на экране (рис. 3) и дважды нажать левую кнопку мыши.

Рис. .3. Пиктограмма текстового редактора Word.

После загрузки на экране появится рабочее окно текстового редактора, представленное на рис. 4.

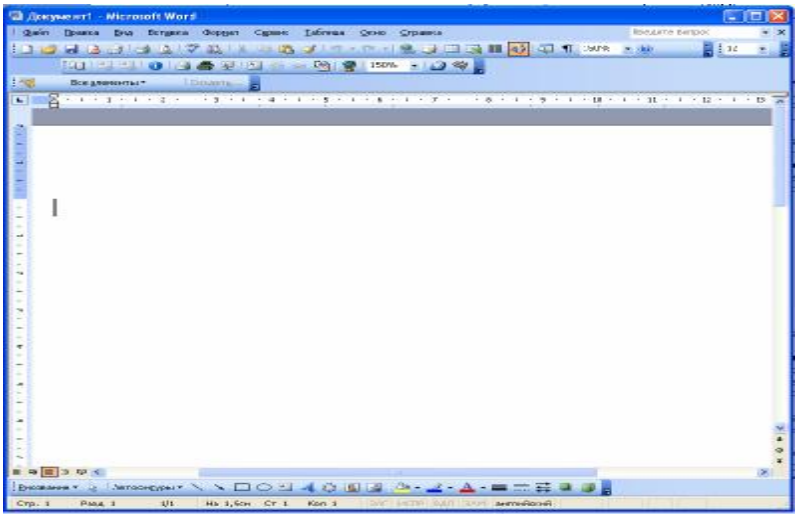


Рис. 4. Рабочее окно текстового редактора Word.

Подготовка текста на ПК в любом текстовом редакторе состоит из двух частей: ввод текста и редактирование текста.

Ввод текста, как правило, осуществляется с помощью клавиатуры, хотя возможны и другие варианты, например, ввод текста с помощью сканера с дальнейшим преобразованием его в файл и редактированием данного файла с помощью текстового редактора. При вводе текста необходимо обратить внимание на переход с русского на латинский регистр. Порядок перехода (комбинация клавиш) задается в Windows.

Word позволяет работать с большим числом различных шрифтов, причем одни из них русифицированы, другие - нет. В связи с этим возможна ситуация, когда не удается перейти с русского на латинский регистр и наоборот. Поэтому прежде всего необходимо подобрать такой шрифт, который, во-первых, удовлетворителен по своему виду (а все шрифты имеют различное изображение символов, в них входящих), во-вторых, русифицирован. Подбор шрифтов осуществляется через Главное меню или с помощью панелей инструментов. Данная процедура одинакова для любого приложения Windows из MS Office. Подробнее процесс выбора шрифта будет описан ниже.

При наборе текста необходимо помнить, что:

указатель мыши отличается от указателя курсора. Обычно указатель мыши выглядит, как стрелка, но если указатель перемещается по части экрана, предназначенной для заполнения текстом, вид указателя меняется на 1-образный;

- указатель курсора всегда находится в текстовом поле документа и имеет вид мигающей вертикальной черты;
- толстая горизонтальная линия в конце набранного текста - это маркер конца текста.

Следующий этап работы с текстом в текстовом редакторе -редактирование набранного текста. Под *редактированием* понимается задание размеров листа, выделение заголовков, задание красной строки в абзацах, вставка рисунков, объектов и другого графического материала в текст. Если текст готовится для представления в гипертекстовом виде, то редактирование включает ввод в текст соответствующих средств в формате HTML. Такие возможности в MS Office 97 имеются.

Рассмотрим набор стандартных функций редактирования на примере Word 97.

Вызов различных функций редактора возможен как с помощью мыши, так и с помощью специальных комбинаций клавиш. Работа с помощью мыши наиболее естественна, но знание некоторых комбинаций «горячих клавиш» полезно для ускорения работы.

Управление редактором, впрочем, как и любым приложением Windows, осуществляется с помощью Главного меню.

Дополнительным средством управления текстовым редактором являются панели: стандартная панель инструментов (рис. 5); панели инструментов редактирования и форматирования (рис. 6) и др.

Рис. 5. Стандартная панель инструментов.

Рис. 6. Панели инструментов редактирования и форматирования:

На эти панели для ускорения работы вынесены кнопки, дублирующие различные действия, выполняемые в текстовом редакторе с помощью опций Главного меню.

При вызове каждого пункта меню на экране появляется подменю, уточняющее действия редактора, которые возможно выполнить при выборе данного пункта меню (рис. 7).

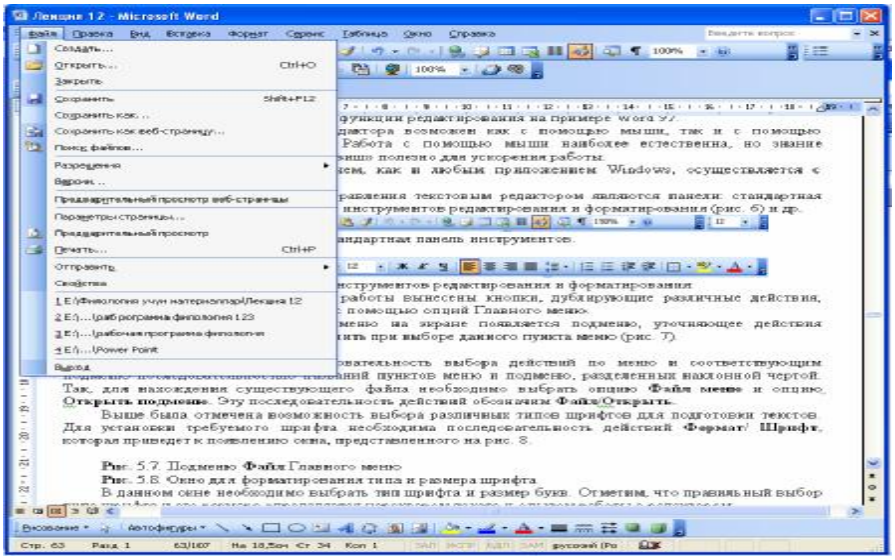


Рис. 7. Подменю **Файл** Главного меню

Условимся обозначать последовательность выбора действий по меню и соответствующим подменю последовательностью названий пунктов меню и подменю, разделенных наклонной чертой. Так, для нахождения существующего файла необходимо выбрать опцию **Файл меню** и опцию **Открыть подменю.** Эту последовательность действий обозначим **Файл/Открыть.**

Выше была отмечена возможность выбора различных типов шрифтов для подготовки текстов. Для установки требуемого шрифта необходима последовательность действий **Формат/ Шрифт,** которая приведет к появлению окна, представленного на рис. 8.

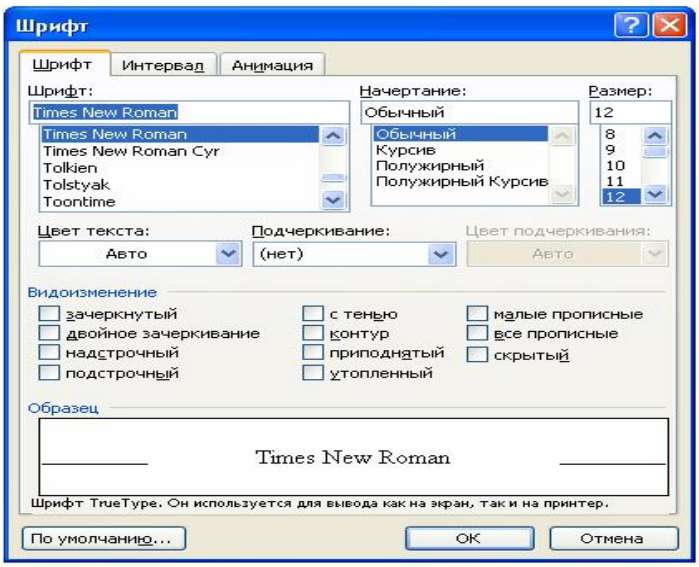


Рис. 8. Окно для форматирования типа и размера шрифта

В данном окне необходимо выбрать тип шрифта и размер букв. Отметим, что правильный выбор типа шрифта и его размера определяется характером текста и опытом работы с редактором.

В настоящее время разработано очень много различных шрифтов. Под *шрифтом* обычно понимают совокупность букв, цифр, специальных знаков, оформленных в соответствии с едиными требованиями. Рисунок шрифта называется *гарнитурой*. Различают шрифты по начертанию (прямой, курсивный), размер шрифта называется *кеглем*. Инсталляция шрифтов осуществляется для Windows.

Поддерживается процедура выбора шрифтов большинством приложений Windows Наиболее интересными являются TrueType-шрифты (в перечне шрифтов они обозначаются буквами TT). Эти шрифты являются масштабируемыми, т.е. без потери качества можно получить любой кегль.

Прежде всего необходимо задать размеры листа бумаги, на котором предполагается печатать текст. Для этого через меню выбирается **Файл/Параметры** страницы. В появившемся меню (рис. 9) необходимо задать в опции **Поля** размеры листа (точнее расстояние от края листа до текста) сверху, снизу, слева, справа. В закладке **Размер бумаги** указываются размер бумаги и его ориентация. Причем ориентация листа альбомная позволяет распределять текст поперек листа. Отметим, что при любой ориентации текста лист бумаги при печати вставляется в принтер обычным образом.

Чтобы произвести с любым фрагментом текста какие-либо операции, следует предварительно отметить или выделить этот фрагмент. Для этого необходимо установить курсор мыши в начало абзаца, нажать и, удерживая левую кнопку мыши, протянуть указатель до конца текста. Выделенный текст будет подсвечен.

Для того чтобы отметить полную строку, достаточно передвигать указатель мыши к левой границе строки до тех пор, пока он не превратится в стрелку, направленную в правый верхний угол. После этого надо нажать левую кнопку. Отметить весь абзац можно, поставив курсор внутри данного абзаца и трижды нажав левую кнопку мыши.

Основа редактирования текста - редактирование заголовков и абзацев. Для редактирования заголовков и абзацев выбирается **Формат/Абзац.** После этого на экране появится окно, представленное на рис. 10.

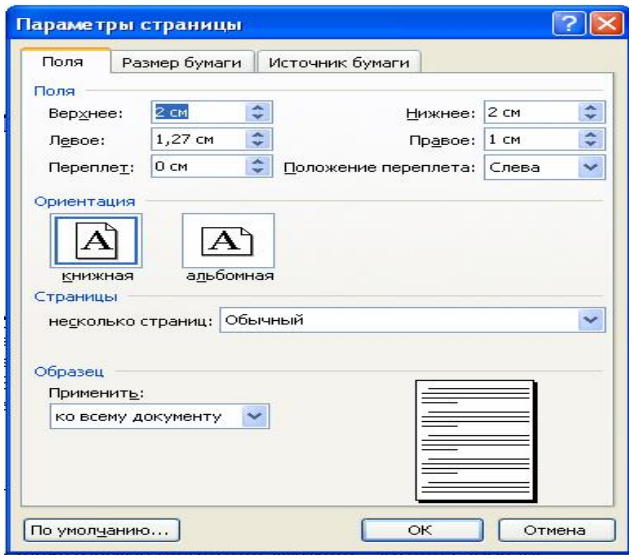


Рис. 9. Окно для установки параметров страницы

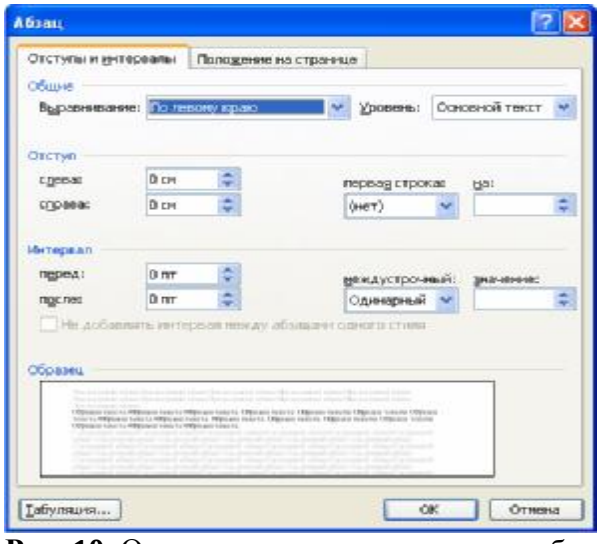


Рис. 10. Окно для задания параметров абзаца.

Заголовок и абзац форматируются одинаково, поэтому заголовок можно считать абзацем специального вида. В окне *Абзац* необходимо задать отступ абзаца справа и слева от границ документа. Пользователь может расположить абзац различным образом в границах текста. Для этого в окне *Выравнивание* необходимо выбрать соответственно **Влево, Вправо, По центру, По ширине** (абзац выровнен по левому краю текста, по правому краю, по центру, по левому и правому краям соответственно).

Для задания расстояния между строками в абзаце следует воспользоваться окном *Межстрочный,* где можно установить одинарный, полуторный, двойной или иной интервал.

Для абзаца существенно наличие красной строки. Для ее задания в окне *Первая строка* выбирается **Красная** и указывается величина отступа - опция Сколько.

Можно задать красную строку, поставив курсор в начале абзаца и нажав клавишу табуляции. Размер перемещения курсора при табуляции можно задавать через линейку, находящуюся под панелями управления. Для того чтобы линейка появилась на экране, следует активизировать ее в пункте меню **Вид.** Если линейка активизирована, надо установить курсор в соответствующее место и нажать левую клавишу мыши. При этом появится специальный знак, который определяет место перехода курсора при нажатии клавиши табуляции.

На линейке может быть несколько символов табуляции. При редактировании часто необходимо удалять фрагменты текста, переносить их в другие места и т.д. Для этого предварительно необходимо отметить часть текста, с которой нужно проводить работу, а затем выбирать опцию **Правка** меню.

При необходимости перенести часть текста в другое место следует отметить переносимую часть текста, выбрать пункт меню **Вырезать,** затем установить курсор в то место, куда надо вставить этот фрагмент, и выбрать последовательность опций **Правка/Вставить.** Аналогично производится операция копирования. Отличие только в том, что вместо **Вырезать** следует выбрать **Копировать.**

Часто при редактировании текста необходимо перемещаться по нему из начала в конец и наоборот. Если текст достаточно велик и притом содержит много графических образов, эта процедура занимает много времени. Для перемещения можно использовать опцию **Правка/Перейти.**

На подменю необходимо выбрать опцию **Страницы** и ввести номер страницы, на которую следует осуществить переход. При этом можно контролировать номер страницы по панели статуса в нижней строке экрана.

Удобным средством поиска является **Закладка.** Эту опцию возможно выбрать на том же подменю. Для того чтобы воспользоваться этим средством, необходимо выбрать **Вставка/Закладка** и на подменю в окне *Имя-Закладки* набрать имя закладки (рис.11).

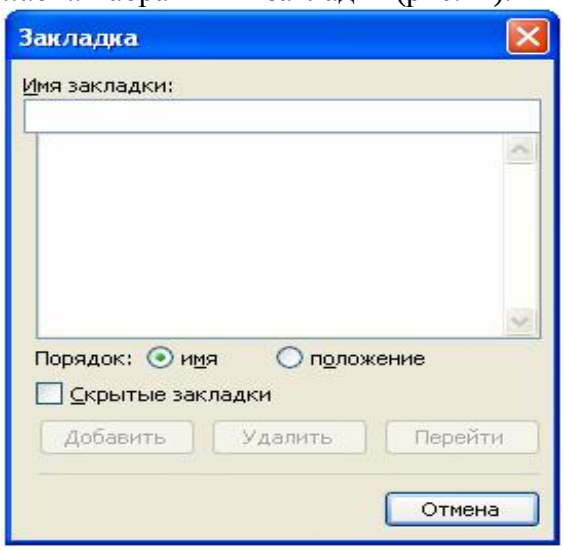


Рис. 11. Подменю **Закладка**

После того как будет введено имя закладки, следует нажать кнопку **Добавить.** Если теперь необходимо переместиться из любого места текста в то место, где установлена закладка, набирается **Правка/Перейти** (затем необходимо выбрать **Объект перехода: Закладка)** или **Вставка/Закладка,** выбирается необходимая закладка (по имени) и нажимается кнопка **Перейти.** Курсор переместится на место закладки. Теперь на этом же подменю достаточно нажать кнопку **Закрыть** и продолжать работу с текстом.

Большинство операций в текстовом редакторе можно проделать, используя панели инструментов, при этом в Word при выборе любой кнопки курсором мыши на экране появляется подсказка функции, выполняемой при нажатии данной кнопки.

Отметим, что действие по выбранной кнопке начинается без дополнительного предупреждения в отличие от выполнения того же действия при выборе последовательности меню. Перечислим наиболее употребительные функции кнопок. На рис. 5 приняты следующие обозначения:

- кнопка 1- для сохранения (записи в файл) текста. Периодическое сохранение текста - хороший стиль работы. Следует взять за правило сохранять текст после набора каждого экрана текста, это предохраняет от потери текста в различных ситуациях, т.е. сберегает время и труд пользователя. Такое же действие оказывает выбор последовательности пунктов меню **Файл/Сохранить;**
- кнопка 2 - для предварительного просмотра текста. Выбрав эту кнопку, можно увидеть, как набранный текст будет выглядеть на бумаге после печати. Аналогичные действия осуществляются после выбора **Файл/Просмотр;**
- выбрав кнопку 3, можно последовательно отменять выполненные действия, т.е. возвращаться к такому виду текста, который был до выбранного действия;
- кнопка 4 служит для вырезания (удаления) отмеченного фрагмента текста. Аналогичные действия выполняются через команды **Правка/Вырезать;**
- кнопка 5 - для копирования отмеченного куска текста. Аналогичные действия реализуются через команды **Правка/Копировать;**
- кнопка 6 - для вставки вырезанного или скопированного текста. Аналогом служат команды **Правка/Вставить.**

На рис. 6 представлены кнопки, служащие для оформления текста:

- кнопки 1 обеспечивают выбор типа шрифта и его размер;
- кнопки 2 позволяют соответственно сделать выделенный предварительно текст, написанный полужирным шрифтом, курсивом, подчеркнутым. Аналогичные действия осуществляются после выбора команды **Формат/Шрифт;**
- кнопки 3 расположат абзац, в котором находится курсор, соответственно выровненным по правому краю; по центру; по правому и левому краям;
- кнопки 4 позволяют подчеркивать (обрамлять) выделенный текст так, как на этих кнопках нарисовано.

Удобным вспомогательным средством является дополнительное меню редактирования. Для его появления следует выделить фрагмент текста и затем нажать *правую* кнопку мыши. Появляющееся контекстно-зависимое меню (рис. 12) позволяет копировать выделенный текст (команда **Копировать),** вырезать его, т.е. копировать с удалением (команда **Вырезать),** вставлять ранее скопированный текст (команда **Вставить),** менять размер букв и стиль их написания (команда **Шрифт),** а также редактировать абзац.

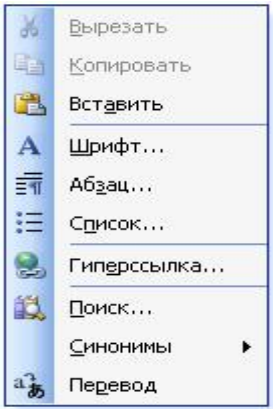


Рис. 12. Контекстно-зависимое меню.

После набора и редактирования текста необходимо его сохранить, для чего следует нажать соответствующую кнопку на панели инструментов или воспользоваться последовательностью действий по меню Файл/Сохранить. Если текст сохраняется первый раз, то файлу, содержащему данный текст, необходимо присвоить имя. Для этого выбирается последовательность действий по

меню **Файл/Сохранить как.** После этого необходимо выбрать имя диска, директорию и ввести имя файла. Созданный файл автоматически получит расширение **doc.**

Необходимо обратить внимание на следующее: Word 97 имеет кодировку символов, отличную от кодировки, принятой в более ранних версиях. Причем возможно прочитать в Word 97 текст, набранный в Word 95/7.0/6.0. Если же текст набран в Word 97, то прочитать его средствами Word 95/7.0/6.0 невозможно.

При работе с одним текстом на машинах с разными версиями офиса целесообразно работать в кодировке Word 95/7.0/6.0, которая понятна в любой версии. Это необходимо учитывать при сохранении текста в Word 97. При первоначальном сохранении текста необходимо выбрать тип файла Word 6/95.

Если пользователь работает с файлом, созданным некоторое время назад, то, чтобы увидеть его на экране, выбирается последовательность действий **Файл/Открыть.** Появляется меню, аналогичное предыдущему, и пользователь выбирает из списка имя требуемого файла, в результате чего этот файл будет загружен.

Текстовый редактор Word обладает широкими возможностями для печати текстов на принтере. Для печати необходимо выбрать последовательность опций **Файл/Печать.**

Открывающееся меню (рис. 13) позволяет:

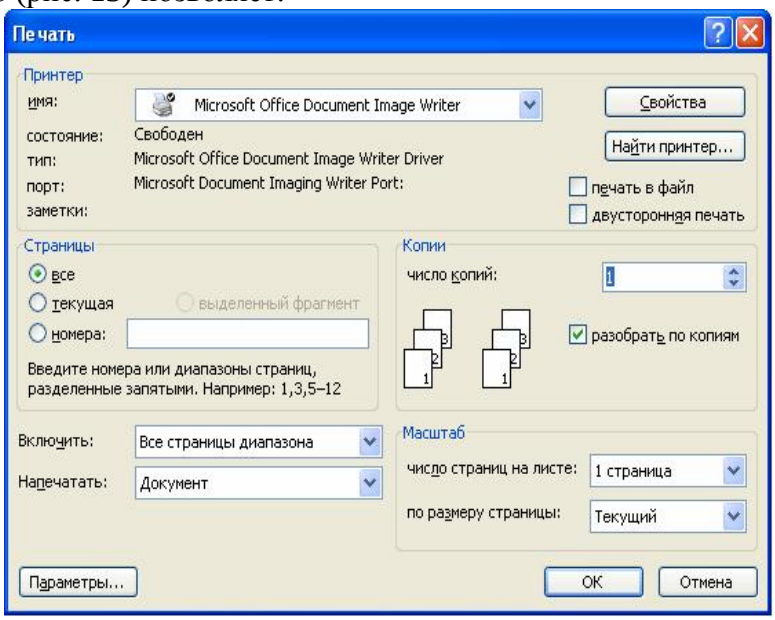


Рис. 13. Окно подготовки печати документа

- задать число копий текста через опцию копий;
- указать - выдать на печать все страницы, только отмеченные, отдельные страницы или последовательности страниц;
- если необходимо распечатать тексты не сразу или на другой машине, то полезно выдать весь текст в специальный файл печати (Печать в файл). После этой операции можно распечатать данный файл на любом принтере.

При подключении к ПК нового принтера необходимо провести определенную работу по настройке редактора на тип этого принтера. В Windows данная работа осуществляется с помощью мастера-установщика принтера.

При печати документа могут возникнуть различные ситуации. В этом случае одно из приложений Windows - Диспетчер печати попросит помощи, выдав на экран соответствующее предупреждение. Отметим, что это сообщение может быть в зависимости от версии Windows как на русском, так и на английском языке.

Возможна ситуация, когда необходимо прекратить процесс печати. В этом случае при нажатии клавиш [Ctrl]+[Esc] появится меню, как при нажатии клавиши [Пуск].

Выбрав на этом меню **Настройка,** а затем на появившемся подменю **Принтеры,** можно удалить задание, т. е. прекратить распечатку текста. Отметим, что если не удалить задание, а выключить принтер, то при последующем его включении печать будет продолжена.

Часто перед печатью потребуется провести некоторую первоначальную обработку текста. Например, вставить номера страниц, дату и время печати и т.д. Такие возможности в редакторе имеются. Для вставки в текст номеров страниц выбираются опции **Вставка/Номера страниц.** На экране появится подменю.

Выбор местоположения номера страницы осуществляется с помощью опций **Положение** (вверху страницы или внизу) и **Выравнивание** (в правом углу страницы, левом или более сложное расположение нумерации на четной и нечетной страницах). Причем в окне *Пример* видно, в каком месте на странице будет находиться номер.

Текущие дата и время устанавливаются с помощью выбора опций **Вставка/Дата и время.**

Текстовый редактор Word имеет развитые средства для обмена информацией с другими приложениями Windows. Кроме того, в этот текстовый редактор встроены средства, которые позволяют рассматривать данный редактор как небольшую издательскую систему.

Редактор имеет средства для вставки в текст готовых и поставляемых вместе с этим редактором рисунков. Задав последовательность опций **Вставка/Рисунок/Картинки**, можно иметь возможность выбрать любой графический объект, рисунок, звуковой или видеофайлы.

Помимо уже готовых рисунков можно изготовить и поместить в текст любые графические образы. Для этого служит графический редактор Paint. Для его вызова необходимо нажать кнопку **Пуск** на экране, затем в появившихся меню выбрать **Программы/ Стандартные/Paint.**

Для переноса информации из Paint в текстовый редактор можно воспользоваться буфером обмена Windows. Для этого в графическом редакторе необходимо выделить информацию, которая должна быть перенесена в текстовый редактор, и поместить ее в буфер операциями **Копировать** или **Вырезать.** После этого необходимо вернуться в текстовый редактор, поставить в необходимое место на экране курсор и выбрать **Правка/Вставить.** Созданный графический образ будет перенесен в текстовый редактор.

Такую возможность - перенос информации через буфер обмена из одного приложения в другое имеют все приложения Windows, но надо иметь в виду, что наличие графических образов в тексте уменьшает скорость работы текстового редактора даже для достаточно мощных компьютеров.

Помимо вставки графических объектов, подготовленных с помощью редактора Paint, Word позволяет вставлять в текст электронную таблицу из пакета Excel 97 и проводить расчеты. После выхода из этого режима в тексте останется таблица Excel, которая в своих ячейках будет содержать те величины, которые были введены или вычислены.

Кроме таблицы из Excel можно воспользоваться таблицей из редактора, которая представляет собой список, состоящий из строк и колонок. Эта таблица является удобным средством для форматирования текста и позволяет выполнять простые расчеты для числовой информации, введенной в эту таблицу.

Для построения таблицы необходимо установить курсор в то место, где нужно поместить таблицу, и затем выбрать последовательность **Таблица/Добавить таблицу.**

В появившемся окне надо указать необходимые характеристики таблицы, т.е. число столбцов и строк, после чего на экране появится таблица с заданным числом строк и столбцов. Следующий шаг - внесение информации в таблицу.

Процесс создания таблицы можно ускорить, если воспользоваться кнопкой **Добавить таблицу** на панели инструментов. После нажатия кнопки появляется решетка, в которой можно задать размер таблицы с помощью мыши.

Появляющаяся таблица разделена на строки и столбцы. Часть таблицы, ограниченная отрезками решетки, называется ячейками. В каждую ячейку может вводиться любая информация. Переход от ячейки к ячейке лучше всего осуществлять клавишей [Tab].

Все манипуляции с таблицей можно проводить, используя опцию **Таблица** Главного меню. Изменить ширину ячейки проще всего мышью Для этого, установив курсор на границу между колонками (курсор изменит свою форму), надо зафиксировать (нажать) левую кнопку мыши и перемещать курсор, увеличивая ширину ячейки.

После заполнения колонки необходимо обозначить границы таблицы, линии, ограничивающие ячейки, и т.д. Для этого следует выделить часть таблицы, в которой нужны ограничительные линии, и воспользоваться кнопками, которые дают все возможные комбинации ограничений.

Более тонкую настройку таблицы необходимо проводить через опцию **Таблица** Главного меню В этой опции можно задавать разный размер колонок, ставить между ними разделители (опция **Добавить таблицу** подменю) и делать многое другое.

Полезная особенность редактора Word - возможность делать колонки. Обычно такое требование предъявляют к настольным издательским системам По сравнению с издательскими системами Word менее приспособлен для таких действий, однако имеющихся в нем возможностей вполне достаточно для пользователя непрофессионала-редактора.

Если текст уже набран, редактор позволит переформатировать его в колонки. Существует и возможность сразу набирать текст в колонках Причем число колонок также может задаваться.

Чтобы разбить текст на колонки, необходимо предпринять следующие действия, нажать кнопку **Колонки** и на появившейся решетке мышью задать число колонок, подведя к ним курсор мыши и зафиксировав левую кнопку.

Полезные возможности предоставляет выбор последовательности опций **Формат/Граница и Заливка.** После выбора данной последовательности опций появится подменю (рис. 14).

В этом подменю можно выбрать то обрамление отмеченного куска текста, которое требуется. Выбрав закладку **Заливка,** можно дать фон любой интенсивности для этого же куска текста.

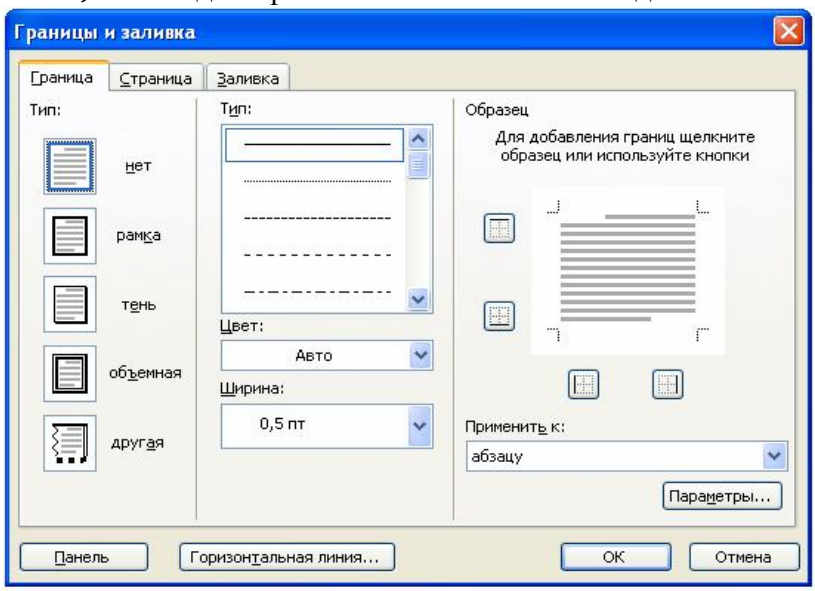


Рис. 5.14. Окно для задания обрамления и заполнения

При работе с большим количеством текстов поиск конкретного текста становится проблемой. Это связано с тем, что поиск текста (файла) происходит по имени, а помнить большое число имен файлов трудно, однако существует удобное средство поиска требуемого файла. Для этого выбирается последовательность **Пуск/Найти/Файлы и папки,** после чего появляется окно (рис. 15).

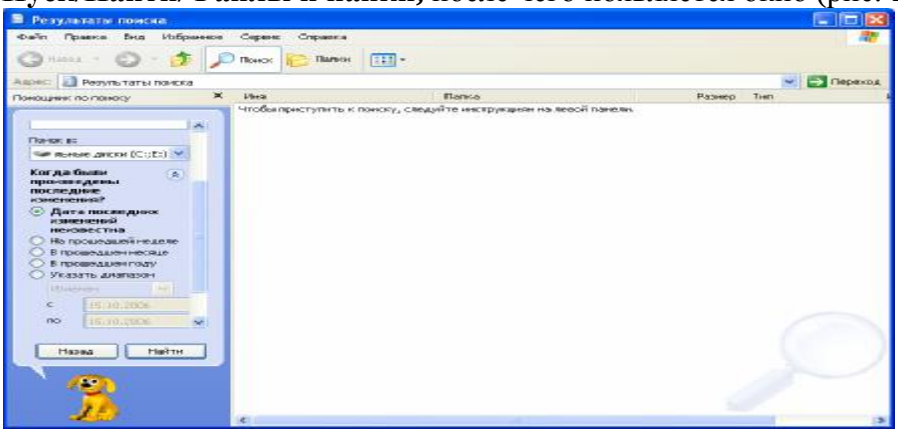


Рис. 15. Окно поиска

В данном окне необходимо указать известные параметры файла, который требуется найти.

Иногда при подготовке текста необходимо создавать и вставлять в него рисунки и схемы, подготовленные автором текста. Для этого служит специальная панель инструментов на экране редактора, которая содержит клавиши, вызывающие соответствующие функции. Если такая панель отсутствует, то необходимо набрать последовательность опций **Вид/Панели инструментов** и на

появившемся меню выбрать бокс **Рисование.** После этого, используя кнопки-инструменты, можно формировать произвольный рисунок.

Конечно, возможности Word не ограничиваются отмеченными выше, а значительно шире. В частности, редакторы позволяют оформлять текстовые фрагменты в виде списков; оформлять сноски; создавать автоматически оглавления; использовать стили и шаблоны для оформления документов; автоматически проверять орфографию и многое другое.

ТАБЛИЧНЫЕ ПРОЦЕССОРЫ

Большинство задач, решаемых в системах организационно-экономического управления, связано с обработкой больших объемов информации, интеграцией данных разных форм и документов, использованием графической интерпретации данных в виде диаграмм и графиков, необходимостью группировки и сортировки данных по разным показателям, проведением анализа данных для дальнейшего принятия решения, а также выводом на печать большого количества отчетных форм. Все эти задачи можно успешно решить, не владея при этом специальными знаниями в области программирования, применив в работе табличные процессоры (электронные таблицы).

В MS Office средством для создания электронных таблиц является табличный процессор Excel. На сегодняшний день наиболее популярны версии Excel 7.0, Excel 97 для Windows 95/98/NT), вышла в свет версия Excel 2000, а также электронные таблицы Quattro Pro фирмы Novell и Lotus 1-2-3 фирмы Lotus Development. Все они работают в среде Windows и выполняют принципиально одни и те же функции с некоторыми различиями в их реализации.

Ниже рассматриваются русскоязычная (локализованная) версия Excel 97 и некоторые новшества, появившиеся в Excel 2000.

Для нормальной работы пакета Microsoft Excel 97 требуются: компьютер с процессором не ниже Intel 486; операционная система Windows 95, Windows 98 или Windows NT; 6 Мбайт оперативной памяти; 18 Мбайт свободного пространства на жестком диске; мышь.

В случае установки версии Excel 2000 минимальные требования к компьютеру несколько выше: процессор Pentium II/III, ОС Windows 95/98/NT, 16 Мбайт оперативной памяти, видеоадаптер SVGA, мышь, CD-ROM.

При этом чем больше объем оперативной памяти, тем выше быстродействие программы.

Excel представляет собой обычное окно в Windows, которое обладает всеми его свойствами, а именно: состоит из рамок, системного меню, заголовка окна, кнопок изменения размеров окна, строки меню, рабочего поля, курсора мыши (рис. 1).

Окно Excel состоит из двух вложенных друг в друга окон. Внешнее - это программное окно Excel, внутреннее окно - это рабочая страница.

Рабочая страница представлена, как правило, в виде таблицы, разграфленной на столбцы и строки. Столбцы обозначены буквами, строки - цифрами. Клетки, из которых состоят электронные таблицы, называются *ячейками*, в них помещают текст, числа, формулы и функции.

Существенным элементом программного окна являются панели инструментов, которые используются для ускорения вызова наиболее часто используемых процедур.

Под панелями инструментов в программном окне располагается строка формул, где находится информация, которая вводится или была введена в рабочую ячейку в виде текста, числа или формулы.

Слева на строке формул показан адрес рабочей (текущей) ячейки. Рабочей ячейкой считается ячейка, в которой в данный момент находится курсор; она помечена серой рамкой.

Ниже рабочего листа располагается строка, называемая *ярлыком* рабочего листа. Здесь показаны названия всех рабочих листов данной рабочей книги. С помощью мыши можно переключаться с одного рабочего листа на другой, а с помощью группы из четырех стрелок, находящихся в левом углу ярлыка рабочего листа, можно переключиться на первый или последний рабочий лист (крайние стрелки) или перемещаться по ярлыкам рабочих листов (средние стрелки).

Самая нижняя строка экрана - *панель статуса*. В ней содержится информация о том, что нужно сделать, чтобы довести выполнение команды до конца.

Первичные операции, которые пользователь производит при работе с табличным процессором, заключаются во вводе данных в таблицу, редактировании, копировании, перемещении и удалении данных.

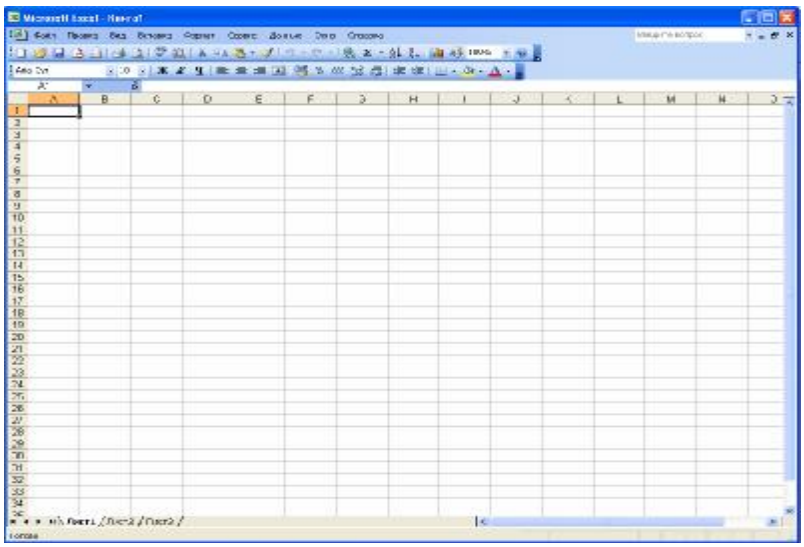


Рис. 1. Рабочее окно Excel

Поскольку Excel является программным продуктом семейства MS Office, многие действия выполняются аналогично действиям в других пакетах этого семейства, таких, как MS Word, MS Access.

К процедурам редактирования относятся: редактирование содержимого ячейки, удаление данных из ячейки, добавление в таблицу столбцов, строк и др.

Для редактирования содержимого ячейки необходимо:

1) сделать эту ячейку текущей (т.е. переместить в нее указатель);
2) нажать клавишу [F2];
3) внести необходимые изменения в содержимое ячейки;
4) нажать клавишу [Enter].

Для удаления данных необходимо:

1) выделить нужный диапазон;
2) нажать клавишу [Delete] (если требуется удалить только содержимое ячеек) или выполнить команду **Правка/Очистить,** которая позволяет предварительно уточнить способ удаления (с форматами или без).

Для добавления в таблицу столбца, строки или группы столбцов (строк) необходимо:

1) выделить количество строк или столбцов, которое нужно добавить в таблицу;
2) выполнить команду **Вставка/Столбцы** или **Строки.**

Новые столбцы будут помещены слева от выделенных, а новые строки - сверху. Названия столбцов и номера строк после выполнения операции добавления автоматически пересчитываются.

К группе команд редактирования также относятся команды изменения ширины столбцов, высоты строк; изменения форматов числовых данных, типов и атрибутов шрифта; подборки и установки цвета; проведения границ и рамок; выравнивания данных; использования специальных стилей. Все эти операции сгруппированы в меню под пунктом **Формат.**

Чтобы начать работать в Excel, необходимо создать новую таблицу или рабочую книгу. Excel автоматически присваивает ей имя Книга с очередным порядковым номером, например Книга 1, и расширением XL. Каждая рабочая книга может состоять из множества листов, которые можно добавлять или удалять по ходу выполнения работы. По умолчанию новая рабочая книга содержит 3 листа. Excel, как и любая электронная таблица, представляет собой инструмент для вычислений. Каждая ячейка в Excel может содержать данные одного из трех типов: текст; число; формула.

При вводе данных они одновременно отражаются в текущей ячейке и строке формул. Чтобы подтвердить завершение операции ввода, достаточно сместить указатель в другую ячейку или нажать клавишу [Enter].

Кроме простых расчетов с использованием арифметических действий Excel позволяет обрабатывать данные с помощью более чем сотни встроенных функций. Функции можно вводить непосредственно в строке формул, т.е. непосредственно в ячейку, однако лучше через диалоговое окно, называемое *Мастер функций.* Признаком того, что в ячейку введена формула, а не текст или простое числовое значение, является *знак равенства.*

Окно *Мастер функций* организовано по тематическому принципу. В левой части окна находятся названия групп, в правой части - функции, принадлежащие к данной группе (рис. 2). Использование окна *Мастер функций* оказывает помощь при задании аргументов функций. В качестве аргументов функции могут использоваться другие функции и адреса ячеек.

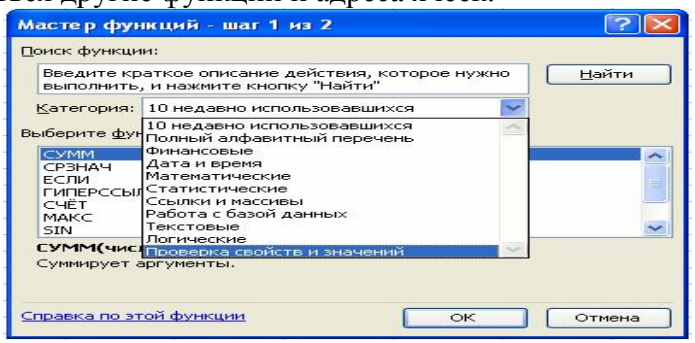


Рис. 2. Окно **Мастер функций**

Наиболее часто используемой функцией является *функция автосуммирования*, поэтому она вынесена на стандартную панель инструментов.

Каждый из трех типов данных может быть представлен в различных форматах. Например, число может быть представлено как целое, с десятичными знаками, в процентах, в денежном формате, в формате даты и времени и др. Кроме того, каждая категория формата предоставляет ряд кодов для преобразования данных. Чтобы задать или изменить формат ячейки, нужно выполнить команду **Формат/Ячейки/Число.**

В появившемся окне *Формат ячеек* (рис. 3) выбирается подходящий формат данных. В стандартной конфигурации Excel 97 на панель форматов ячеек вынесены пять наиболее часто употребляемых форматов чисел: *денежный, процентный, разделитель тысяч, увеличить разрядность, уменьшить разрядность.*

При необходимости на панель инструментов можно вывести дополнительные элементы форматирования.

Для удобства работы в Excel имеется возможность присваивать имена отдельным ячейкам или областям, которые затем можно вводить в формулы наравне с адресами. При этом ячейкам или областям автоматически присваивается имя, составленное из текста, который введен в ячейку, расположенную над полем или слева от него. Разумеется, это имя может быть изменено по усмотрению. Имена присваиваются с помощью диалогового окна *Присвоение имени,* выполнив последовательность действий: **Вставка/ Имя/ Присвоить.**

При присвоении имен нужно учитывать следующее:

- имена должны состоять из букв, цифр, точек и символов подчеркивания;
- пробелы не допускаются;
- прописные и строчные буквы воспринимаются одинаково;
- можно использовать как латинский, так и русский регистры.

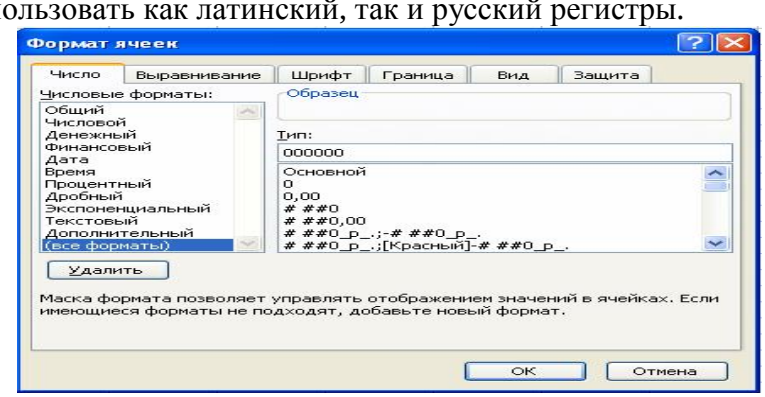


Рис. 3. Окно форматирования ячеек

Имя поля используется при абсолютном обращении к ячейке, т.е. когда значения берутся из ячейки с точно указанным адресом. Превратить адреса в абсолютные можно и другим способом, а именно пометив поля значком «$» (доллар).

В левой части строки формул находится список имен полей, созданный с помощью окна *Определить имя.* При выборе какого-либо имени из этого списка будет осуществлен автоматический переход в заданную ячейку.

Существует и другой тип обращения к ячейке - относительный. В этом случае положение нужной ячейки определяется относительным расстоянием до нее от ячейки ввода. Если при этом меняется положение ячейки ввода, то меняются и адреса ячеек, из которых берутся данные.

Excel дает возможность защитить данные от случайного или намеренного изменения. Файл целиком можно защитить с помощью сервисной функции *Защита.* Файл будет защищен паролем, не введя который, нельзя будет произвести ни одного изменения в таблице. Наряду с этим существует возможность закрыть для доступа только часть ячеек. Для этого та часть ячеек, которая остается открытой, выделяется и в диалоговом окне *Формат ячеек*, появившемся после выполнения последовательности действий **Формат/Ячейки / Защита,** щелчком мыши выключается поле *Защищаемая ячейка*, а затем с помощью сервисной функции *Защита* задается пароль. Пароль состоит из набора букв и цифр, но на экране вместо них появляются символы - звездочки. Снять блокировку можно, введя верный пароль в окне *Снять защиту*.

Одной из важных функций, заложенных в Excel, является создание внутри электронной таблицы баз данных. Электронная база данных есть некоторое хранилище информации, определенным образом рассортированной и разложенной по особым категориям и признакам. С помощью функции Excel, обслуживающей базы данных, можно сохранить данные в специальной форме, выбрать из них требуемые сведения, обработать, распечатать и т.п. Работа с такими базами данных не требует специальных знаний, для этого достаточно использовать общие правила работы с электронными таблицами. Единственное, что нужно сделать, - это выделить интересующую часть таблицы, так называемый интервал базы данных, присвоить ему имя, и впоследствии при обращении к этому имени база данных будет выделена пунктиром.

Для работы с базами данных используется пункт меню **Данные.** Информацию базы данных можно сортировать в алфавитном порядке, по возрастанию или убыванию числовых значений с помощью пункта **Консолидация,** находить и отбирать с помощью пункта **Фильтр,** уничтожать и распечатывать необходимую информацию.

Важным инструментом при работе с базами данных является использование критериев отбора. В состав критериев отбора могут входить числа, метки, формулы, логические операторы и др.

Работая с базой данных, удобно использовать опцию формы данных **Форма. В** результате на экране появится диалоговое окно, позволяющее легко просматривать, редактировать и осуществлять поиск по заданному критерию.

Возможность Excel интерпретировать данные в графическом виде делает работу с таблицами более наглядной и эффективной. В Excel на экране может одновременно присутствовать как таблица, так и диаграмма. После вывода диаграммы на экран пользователь имеет возможность оперативно вносить изменения в таблицу, что мгновенно автоматически отражается на диаграмме.

Создается внедренная диаграмма с помощью **Мастера диаграмм,** который представлен на стандартной панели инструментов значком  .

Перед созданием диаграммы нужно отметить участок таблицы, информация с которого должна быть показана на диаграмме. Затем нужно щелкнуть мышью на кнопке **Мастер диаграмм,** в результате чего курсор приобретет форму маленького черного креста. Поместив крест на выбранное для диаграммы место, перемещением мыши надо растянуть прямоугольник, после чего программа приступает к пошаговому приглашению пользователя по построению диаграммы. Обычно построение диаграмм осуществляется в пять шагов. На каждом шаге выполняются определенные действия, и в результате получается ожидаемый результат.

Excel облегчает работу с электронными таблицами, обеспечивая просмотр данных в следующих режимах:

- параллельный просмотр областей одной таблицы;
- параллельный просмотр нескольких файлов одновременно;
- просмотр с изменением масштаба;
- просмотр таблицы с прокруткой и фиксацией отдельных строк и столбцов.

Режим *Параллельного просмотра областей одной таблицы* позволяет одновременно видеть различные (или перекрывающиеся) части одной и той же таблицы. Например, при вводе больших таблиц можно разбить экран на области, а именно в верхней части оставить названия колонок, а в нижней можно сколь угодно долго уходить вниз, и всегда можно видеть, в какой колонке работает пользователь. Кроме того, можно зафиксировать столбец, и это позволит точно знать, какой смысл имеет то или иное число в каждой строке.

Разбиение рабочего листа на области можно выполнить двумя способами: с помощью меню и с использованием мыши. В первом случае надо выбрать команды **Окно/Разделить.** Если установить курсор мыши на среднюю точку пересечения малых окон, то, удерживая нажатой левую кнопку мыши, можно перемещать точку разделения по экрану, изменяя размеры окон или вообще убирая их с экрана. Во втором случае необходимо установить курсор мыши на маленький темный штрих выше стрелки, которая направлена вверх в правой полосе прокрутки. Курсор мыши превратится в графический символ, который имеет вид знака равенства с перпендикулярной вертикальной стрелкой с двумя остриями. Нажав левую кнопку мыши и удерживая ее нажатой, следует передвинуть разделитель экрана вниз. Экран поделился на два горизонтальных окна. Для разделения экрана по вертикали необходимо установить курсор мыши на соответствующий штрих в правом углу нижнего поля прокрутки и передвинуть разделитель экрана влево.

Переходить из окна в окно можно с помощью мыши, а также нажимая на клавишу [F6] или [Shift]+[F6].

В режиме *Параллельного просмотра нескольких файлов одновременно* все активные файлы могут быть одновременно представлены на экране в отдельных окнах. Это выполняется командой **Окно/Упорядочить** все.

Для фиксации части рабочего листа надо выполнить команды **Окно/Фиксировать подокна.**

Важный этап работы с табличным процессором - оформление полученных таблиц, которое осуществляется аналогично другим пакетам семейства MS Office, например Word.

С помощью панели инструментов *Форматирование* можно изменить шрифт, стиль, размер символов для выделенного фрагмента, выполнять обрамление и изменять цвет строк и столбцов.

Те же действия можно выполнить, используя пункт меню **Формат/Ячейки.** Для изменения шрифта выбирается закладка **Шрифт,** для обрамления - **Граница,** для изменения цвета - **Вид,** для выравнивания текста - **Выравнивание.**

После того как таблица создана и оформлена, ее можно распечатать, причем предварительно целесообразно просмотреть на экране изображение напечатанной страницы. Для этого нужно воспользоваться общей для всех прикладных программ семейства MS Office кнопкой **Просмотр печати** на стандартной панели инструментов или выполнить команду **Файл/ Просмотр.**

Далее необходимо выяснить, какой именно принтер подключен к компьютеру и, исходя из его технических возможностей, определить формат печати, способ подачи бумаги, размер бумаги и др. Все эти установки можно сделать, выполнив команды **Файл/ Печать** или **Файл/Параметры страницы** (рис. 4).

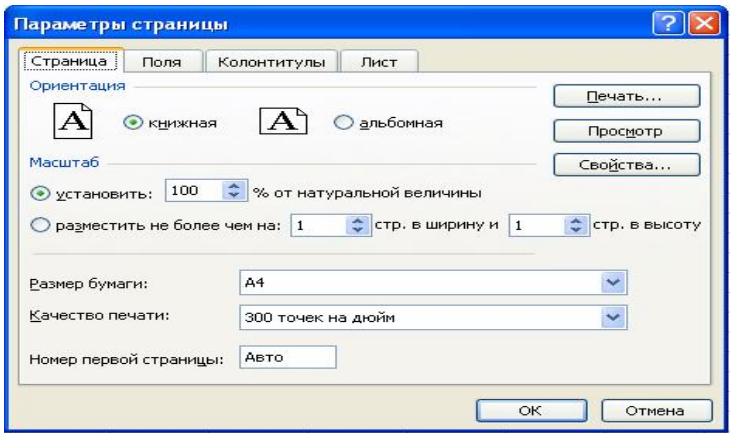


Рис. 4. Окно для установки параметров печати

Находясь в диалоговом окне *Параметры страницы*, можно также выполнить следующие действия: изменить расположение текста относительно начала листам;

- пронумеровать страницы;

- изменить отступы от края листа до текста;
- выделить область печати;
- изменить порядок страниц при печати;
- увеличить или уменьшить размер рабочей страницы при выводе на печать (изменение размеров распечатки не влияет на реальный масштаб рабочего листа);
- создать верхний и нижний колонтитулы;
- поместить текущую дату и время в колонтитул;
- ввести названия строк и столбцов для печати;
- вывести на печать или убрать линии сетки;
- просмотреть список шрифтов, которые доступны в данный момент для распечатки документа, и в соответствии с ним изменить шрифты заголовков в таблицах.

Следует обратить внимание на следующие возможности Excel 97.

Автозамена позволяет запомнить некоторый текст под каким-либо именем. Затем, введя установленное имя, программа подставит вместо него нужный текст.

Установив режим *автоввода*, Excel по первым буквам, введенным в ячейку, предложит автоматически закончить ввод всего слова.

Функция *автовычисления* (автокалькулятор) позволяет увидеть результат промежуточного суммирования в строке состояния, выделив определенные ячейки таблицы и указав, какого типа результат желательно получить - сумму, среднее арифметическое или значение счетчика, отражающего количество отмеченных элементов.

Режим *автоматической фильтрации* позволяет быстро производить выборки из записей таблицы по заданным критериям.

Кроме того, есть возможность работы с *географической картой* для визуального анализа данных по географическим регионам.

Вышеперечисленные приемы работы с электронными таблицами можно отнести к общим, единым для всех версий Excel. Каждая последующая версия дополняется новыми возможностями, вызванными необходимостью улучшить, сделать более понятным, наглядным интерфейс программы, исправить обнаруженные недочеты, а также требованиями времени, связанными с возникновением новых технологий. Так, в электронном процессоре Excel 97 появились дополнительные средства для использования гипертекстовых, мультимедийных, графических, а также сетевых технологий. Особенно актуально это в связи с тем, что развитие экономики и бизнеса немыслимо без использования современных информационных технологий, в частности сети Интернет, а электронные таблицы являются подходящим инструментом для решения подобных задач.

Появилась возможность сохранения данных, представленных в форме листа книги Microsoft Excel, в формате языка HTML (HyperText Markup Language). Для просмотра таких документов используется средство просмотра Web (например, Microsoft Internet Explorer). Чтобы иметь возможность получать ответные сообщения от пользователей узла Web, можно создать в Microsoft Excel форму, предназначенную для сбора данных, вводимых пользователями.

Приведенные возможности получили дальнейшее развитие в версии Excel 2000, являющейся на сегодняшний день последней разработкой компании Microsoft в этой области. В ней еще больше усовершенствованы средства коллективной работы и расширены возможности публикации документов в Интернете. В состав Microsoft Excel 2000 входят средства, позволяющие работать с корпоративными данными, проводить их анализ, формировать сводные таблицы и на их основе получать отчеты.

Значительное внимание в Excel 2000 уделяется работе с диаграммами как средству для анализа данных. Появилось новое понятие "сводная диаграмма" (Pivot Chart). Эта диаграмма позволяет представить информацию сводной таблицы (Pivot Table) в удобной графической форме. При изменении сводной таблицы Pivot Chart меняется автоматически. В предыдущих версиях эта процедура была затруднена и требовала ручного вмешательства пользователя. Добавлены некоторые новые типы диаграмм - шаговые, трехмерные комбинированные.

Следует подчеркнуть, что Excel 2000 с помощью нового формата денежных единиц обеспечивает поддержку новой европейской валюты евро и допускает ее отображение как в виде символа евро, так и в виде трехбуквенного кода ISO (EUR). Благодаря этому пользователи электронных таблиц в Ёвропе

и Соединенных Штатах, которые ведут дела с европейскими компаниями или правительствами, могут обмениваться данными с самыми современными значениями валют.

К улучшениям в пользовательском интерфейсе Excel 2000 можно отнести способ отображения выделенных ячеек не в инвертированном виде, а лишь слегка затененными, что позволяет увидеть результаты вносимых изменений, а также элементы форматирования ячеек, не снимая выделения. Кроме того, в некоторых ситуациях Excel 2000 изменяет вид курсора, помогая пользователю сориентироваться в дальнейших действиях.

Обращает на себя внимание и улучшенный буфер обмена при работе с Excel 2000. В предыдущих версиях буфер обмена очищался после выполнения любой операции. Теперь можно копировать данные из буфера столько раз, сколько нужно.

Основные термины

Копирование данных, экспресс-меню, текстовый редактор, пиктограмма, меню, панели инструментов, буфер, ячейка, панель статуса, мастер функций, присвоении имени, параметры страницы, автозамена.

К о н т р о л ь н ы е в о п р о с ы

1. Что понимается под пакетами электронного офиса?
2. В чем заключаются общие принципы работы программных продуктов MS?
3. Что понимается под контекстным меню?
4. Назовите основные функции текстового редактора WORD.
5. В чем разница между объектами типа «рисунок» и «кадр»?

Тема 11. Прикладные программные средства офисного назначения. Программы подготовки презентаций.

План:

1. Общая характеристика ППП POWER POINT.
2. Создание новой презентации.
3. Создание анимации слайдов.
4. Планирование демонстрации слайдов и настройка временных интервалов для демонстрации слайдов.
5. Запуск и управлении демонстрацией слайдов.

ПРОГРАММЫ ПОДГОТОВКИ ПРЕЗЕНТАЦИЙ ОБЩАЯ ХАРАКТЕРИСТИКА ППП POWER POINT

С помощью данного программного средства пользователь может самостоятельно подготовить выступление с использованием компьютерных слайдов, которые можно представить на экране или распечатать на прозрачной пленке, демонстрировать обычным образом на экране ПК либо создать конспект выступления и материал для раздачи слушателям.

ППП Power Point создает файл с расширением .ppf, содержащий набор слайдов. Программа предоставляет пользователю значительное разнообразие шаблонов слайдов на различные темы. *Шаблоны* - это пустые слайды с размещенными в них заглушками, в которые можно вставить свой текст, рисунки, таблицы и диаграммы. Можно изменить цветовое и художественное оформление любого шаблона презентации, выбрав дизайн по своему усмотрению и снабдив каждый слайд особыми эффектами.

После загрузки Windows, используя Главное меню, программу **Проводник** или пиктограмму этого ППП, запускается Power Point.

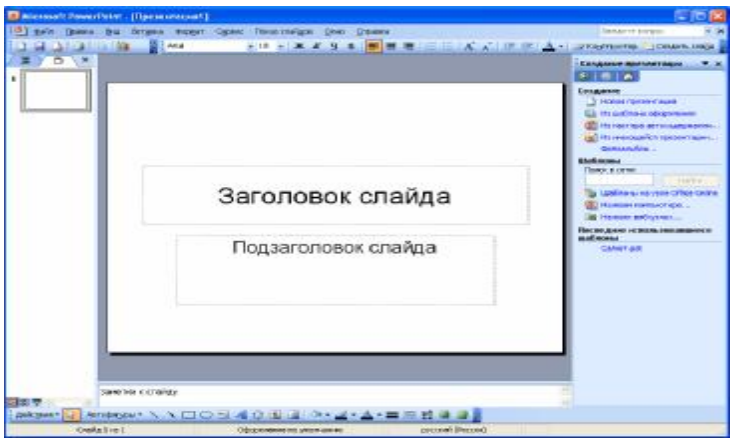


Рис. 1. Окно программы **Power Point**

При этом на экране появляются Главное окно и окно программы (рис. 1), по которому пользователь выбирает способ построения презентации (с помощью Мастера автосодержания, по предлагаемому шаблону, пустую презентацию) либо открывает ранее созданную презентацию.

В верхней части Главного меню, как и для рассмотренных ранее офисных программ, расположены строка меню, панели инструментов, панель режимов работы.

Строка меню содержит стандартные пункты: **Файл, Правка, Вид, Вставка, Формат, Сервис, Рисунок, Окно, Справка,** команды которых почти аналогичны командам меню других офисных программ.

В частности, в меню Файл содержатся следующие команды: **Создать**..., **Открыть..., Закрыть..., Сохранить, Сохранить как..., Упаковка..., Свойства, Параметры слайда..., Печать. В** меню **Правка** пользователь также видит знакомые команды: **Вырезать, Копировать, Вставить, Специальная вставка, Очистить, Выделить все, Найти, Заменить, Перейти** и др.

СОЗДАНИЕ НОВОЙ ПРЕЗЕНТАЦИИ

Новую презентацию можно создавать, используя предлагаемые программой способы: Мастер автосодержания, шаблон презентации, пустая презентация.

Первый способ обеспечивает достаточно быструю технологию создания новых презентаций, предлагая некоторую стандартную структуру презентации и текст.

Для создания презентации этим способом необходимо выполнить следующие операции.

1. В меню Файл_выбрать команду **Создать.**
2. На вкладке **Презентации** диалогового окна *Создать презентацию*_выбрать «Мастер автосодержания».
3. Нажать клавишу [OK].
4. Нажимая кнопки **Далее, Назад, Отмена** и **Готово** и выполняя предписываемые действия, создавать презентацию.

При создании презентации в режиме пустой презентации пользователь после загрузки программы выбирает пункт **Пустая презентация** и выполняет следующую последовательность действий:

- в открывшемся окне *Создать слайд* выбирает из предлагаемых видов авторазметки слайдов устраивающий его и нажимает клавишу [OK];
- в появившиеся поля вставляет свой оригинальный текст или необходимые объекты;
- для заполнения следующего слайда нажимает кнопку **Создать слайд** или выбирает команду **Вставка/Создать слайд.**

Создаваемая презентация будет использовать цветовую гамму, заголовок и стили текста презентации, установленные по умолчанию в Power Point, либо к ней применяется Оформление по умолчанию (оно находится в нижней части окна) одного из предлагаемых в окне диалога *Дизайны презентаций вида оформления* (например, **Ленты, Метеор, Тетрадь и др**.).

При желании пользователя создавать презентации с помощью шаблона дизайна для оформления ее в едином стиле после загрузки программы Power Point необходимо сначала выбрать пункт меню **Файл,** а в нем команду **Создать.** После этого на вкладке **Презентация** выбрать шаблон дизайна (например, Финансовый отчет компании) и нажать клавишу [OK]. Нажав затем на панели

инструментов кнопку **Создать слайд** и выбрав нужный для пользователя слайд, он продолжает создавать презентацию по описанной выше технологии.

Для создания собственного нестандартного шаблона дизайна (специального формата, цветовой схемы) и применения его к любой презентации пользователь должен выполнить следующую последовательность операций.

1. В меню **Файл** выбрать команду Создать.
2. В появившемся окне *Создать презентацию* найти вкладку **Дизайны презентации** и из нее выбрать устраивающий его шаблон (например, Водоворот) и щелкнуть на кнопке ОК.
3. Выбрав авторазметку слайда и заполнив его необходимой информацией, можно изменить цветовую схему слайда. Для этого выполняется такая последовательность действий:

- в меню **Формат** выбирается команда **Цветовая схема слайда;**
- в открывшемся окне выбирается вкладка **Специальная;**
- в группе **Цвета схемы** выбирается цвет элемента (фон, текст и линии, заголовок, заливка и т.д.), который желательно изменить, после чего нажимается кнопка **Изменить цвет;**
- используя вкладки **Обычные** или **Спектр,** выбирается нужный цвет, оттенок, насыщение, яркость, и нажатием кнопки ОК обеспечивается его применение к слайду.
- Для добавления или изменения элемента фона слайда, используя образец слайда и новый образец заголовка, достаточно выполнить такой набор действий:
- в меню **Вид** выбрать команду **Образец,** а затем **Образец слайдов;**
- добавляем, щелкнув внутри слайда, объект, например, текст заголовка, содержательный текст или иной объект;
- для возврата **к** исходному слайду выбирается команда **Слайды** из меню Вид.

Сохранить созданный шаблон можно, выполнив команду **Сохранить как**...из меню **Файл,** введя имя для своего шаблона дизайна и щелкнув **Шаблоны презентаций** в окне *Тип файла*.

При создании презентаций пользователь может добавлять текст в созданные слайды, проводить его корректировку, включать в слайд в различных форматах дату и время, добавлять номер слайда, колонтитул, логотипы, включать готовые картинки или рисовать свои собственные, вставлять таблицы, диаграммы и редактировать их, проверять орфографию и многое другое.

Упростить включение в слайды готовых фигур (кубов, цилиндров, ромбов, фигурных стрелок, элементов, блок-схем, выносок, соединительных линий и др.) можно с помощью кнопки **Автофигуры** инструментальной панели *Рисование*. Перемещая готовые фигуры по слайду, внося в них текст, поворачивая с помощью кнопки **Свободное вращение,** можно дополнительно усилить эффекты.

СОЗДАНИЕ АНИМАЦИИ СЛАЙДОВ

Презентация может демонстрироваться, как показ некоторой очередности слайдов. Однако для повышения качества презентации, придания ей большего визуального эффекта целесообразно спланировать для каждого слайда в отдельности и составляющих его элементов способы появления каждого такого маркированного пункта или графики, например полет слева, всплытие снизу, падение сверху и т.д. Текст в слайде может появляться по абзацам, по словам, по буквам. Можно также задавать последовательное появление графических изображений и других объектов, таких, как диаграммы, клипы и пр. Подобные возможности устанавливаются командами из меню **Показ слайдов:** ***Встроенная анимация, Настройка анимации, Переход слайда*** (рис. 2).

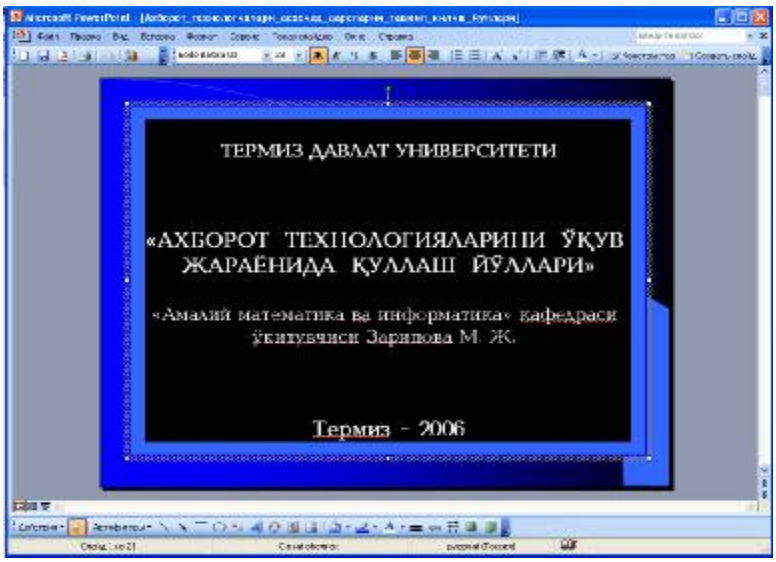


Рис. 2. Окно с меню

Рассмотрим пример, как задать порядок появления объектов в слайде. В режиме слайдов следует выделить текст или графический объект, который должен появиться первым. После этого в меню **Показ слайдов** выбирается команда **Настройка анимации,** в диалоговом окне *Настройка анимации* выбирается вкладка **Эффекты** (рис. 3). Далее в окнах выбора эффекта делаются нужные установки. Например, задается очередность появления в слайде объектов (заголовка, рисунка, текстовой части и др.); появления объекта (вылет сверху, растворение, спираль и т.д.), появления текста (по буквам, словам либо целиком); установки по цвету после анимации и др.

Такая же операция повторяется по очереди для каждого объекта, к которому применяются эффекты.

Можно назначить эффекты для слайдов и включенных в них объектов, если предварительно выбрать в меню **Вид** команду **Сортировщик слайдов.** При этом на экране монитора разместится весь набор слайдов в уменьшенном виде. Щелкнув по слайду, для которого необходимо настроить эффекты, в окнах, расположенных под панелями инструментов, производится выбор необходимых эффектов. Кроме того, в режиме ***Сортировщика*** *слайдов* можно менять местами слайды, определяя их последовательность при показе (демонстрации). Для этого достаточно щелкнуть на слайде, и при нажатии левой клавиши мыши отбуксировать слайд в новую позицию (в новое место).

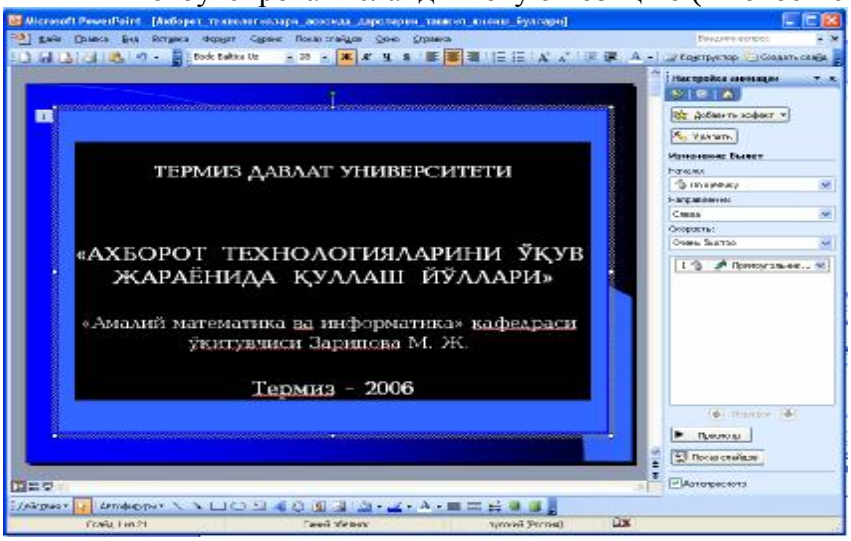


Рис. 3. Окно установки эффектов

ПЛАНИРОВАНИЕ ДЕМОНСТРАЦИИ СЛАЙДОВ И НАСТРОЙКА ВРЕМЕННЫХ ИНТЕРВАЛОВ ДЛЯ ДЕМОНСТРАЦИИ СЛАЙДОВ

При демонстрации слайдов главным является их содержание. Но для усиления влияния на пришедших на презентацию лиц целесообразно, чтобы использованные приемы построения слайдов, появление их, переходы, различные анимационные эффекты помогали слушателям акцентироваться на тех или иных пунктах. Например, учитывая, что человек читает слева направо, целесообразно так

сконструировать слайды, чтобы сообщения «влетали» слева. Для подчеркивания какого-либо важного сообщения можно обеспечить его появление справа или по отдельным буквам слева или справа. Такой прием может привлечь большее внимание слушателей и придать этому пункту большую значимость.

На реакцию аудитории существенно влияют скорость появления необходимой информации, фиксированное нахождение ее на экране, чтобы можно было прочесть сообщение. Поэтому при подготовке презентации используются возможности Power Point по выставлению скорости и времени демонстрации отдельных слайдов.

Величина интервала времени, в течение которого слайд должен оставаться на экране во время демонстрации, устанавливается вручную либо путем использования возможности репетиции в Power Point, которая позволяет устанавливать интервалы автоматически, регулируя их вручную, и проводить репетицию (тренировку) еще раз, чтобы изменить время показа слайда.

Установка временных интервалов для демонстрации обеспечивается последовательностью следующих процедур.

- Выделение в режиме *Сортировщик слайдов* (меню **Вид)** слайда.
- Щелчок по значку Переходы.
- В окне *Переход* слайда установить продвижение Автоматически и ввести величину интервала (в секундах), в течение которого слайд должен оставаться на экране (рис. 4).
- Повторить эту процедуру для каждого слайда, для которого временные интервалы устанавливаются вручную.
- Для установки временных интервалов во время репетиции необходима следующая очередность выполнения процедур:
- в меню **Показ слайдов** выбрать команду **Настройка времени** и в появившемся окне *Настройка времени* нажать среднюю кнопку **П**, когда произойдет отсчет времени, необходимый для демонстрации слайда;
- для переустановки времени можно воспользоваться кнопкой **Повторить** этого же окна;
- для завершения установки времени для данного слайда следует нажать на кнопку закрытия окна.

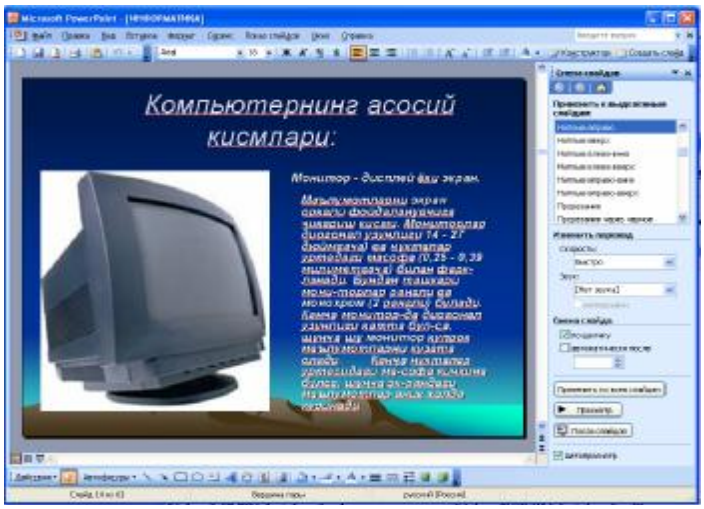


Рис. 4. Окно смена слайдов

ЗАПУСК И УПРАВЛЕНИЕ ДЕМОНСТРАЦИЕЙ СЛАЙДОВ

Демонстрировать подготовленную презентацию можно в двух режимах: вручную и в автоматическом режиме. В первом случае слайды меняются путем нажатия на клавишу мыши, и каждый переход - это смена кадра.

При режиме автоматической демонстрации презентации смена слайдов (кадров) происходит на основе установленных для каждого слайда временных интервалов. Для задания автоматической демонстрации презентации необходимо в меню **Показ слайдов** выбрать команду **Показ** либо нажать внизу экрана кнопку **Показ** слайдов.

В слайды можно включить кнопки управления (меню **Показ слайдов** - управляющие кнопки) и гиперссылки для перехода к предыдущему слайду, к первому или последнему, к слайду с определенным номером и т.д.).

Основные термины

Шаблоны, создать презентацию, создать слайд, дизайны презентаций, анимации слайдов, сортировщик слайдов, настройка времени, показ слайдов.

К о н т р о л ь н ы е в о п р о с ы

1. Перечислите способы создания новых презентаций.
2. Как создать нестандартный шаблон дизайна презентации?
3. Что такое анимация слайдов и как она создается?
4. Что такое анимация?
5. Что такое разметка слайдов?
6. Как вставить в презентацию звук?
7. Как делаются заметки в ходе презентации?

Тема 12. Технология подготовки и решения задач с помощью компьютера.

План:

1. Какие этапы включает в себя решение задач с помощью компьютера?
2. Математическая модель.
3. Основные этапы процесс разработки программ.
4. Отладка и тестирование.
5. Тестовые данные.

Какие этапы включает в себя решение задач с помощью компьютера?

Решение задач с помощью компьютера включает в себя следующие основные этапы, часть из которых осуществляется без участия компьютера.

1. Постановка задачи:

- сбор информации о задаче;
- формулировка условия задачи;
- определение конечных целей решения задачи;
- определение формы выдачи результатов;
- описание данных (их типов, диапазонов величин, структуры и т.п.).

2. Анализ и исследование задачи, модели:

- анализ существующих аналогов;
- анализ технических и программных средств;
- разработка математической модели;
- разработка структур данных.

3. Разработка алгоритма:

- выбор метода проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы, псевдокод и др.);
- выбор тестов и метода тестирования;
- проектирование алгоритма.

4. Программирование:

- выбор языка программирования;
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования.

5. Тестирование и отладка:

- синтаксическая отладка;
- отладка семантики и логической структуры;
- тестовые расчеты и анализ результатов тестирования;

- совершенствование программы.

6. Анализ результатов решения задачи и уточнение в случае необходимости математической модели с повторным выполнением этапов 2 - 5.

7. Сопровождение программы:

- доработка программы для решения конкретных задач;
- составление документации к решенной задаче, к математической модели, к алгоритму, к программе, к набору тестов, к использованию.

Что называют математической моделью?

Математическая модель — это система математических соотношений — формул, уравнений, неравенств и т.д., отражающих существенные свойства объекта или явления.

Всякое явление природы бесконечно в своей сложности. Проиллюстрируем это с помощью примера, взятого из книги В.Н. Тростникова "Человек и информация" (Издательство "Наука", 1970).

... Обыватель формулирует математику задачу следующим образом: *"Сколько времени будет падать камень с высоты 200 метров?"* Математик начнет создавать свой вариант задачи приблизительно так: *"Будем считать, что камень падает в пустоте и что ускорение силы тяжести 9,8 метра в секунду за секунду. Тогда ..."*

— *Позвольте,* — может сказать "заказчик", — *меня не устраивает такое упрощение. Я хочу знать точно, сколько времени будет падать камень в реальных условиях, а не в несуществующей пустоте.*

— *Хорошо,* — согласится математик. — *Будем считать, что камень имеет сферическую форму и диаметр... Какого примерно он диаметра?*

— *Около пяти сантиметров. Но он вовсе не сферический, а продолговатый.*

— *Тогда будем считать, что он имеет форму эллипсоида с полуосями четыре, три и три сантиметра и что он падает так, что большая полуось все время остается вертикальной. Давление воздуха примем равным 760 мм ртутного столба, отсюда найдем плотность воздуха...*

Если тот, кто поставил задачу на "человеческом" языке не будет дальше вмешиваться в ход мысли математика, то последний через некоторое время даст численный ответ. Но "потребитель" может возражать по-прежнему: камень на самом деле вовсе не эллипсоидальный, давление воздуха в том месте и в тот момент не было равно 760 мм ртутного столба и т.д. Что же ответит ему математик?

Он ответит, что точное решение реальной задачи вообще невозможно. Мало того, что форму камня, которая влияет на сопротивление воздуха, невозможно описать никаким математическим уравнением; его вращение в полете также неподвластно математике из-за своей сложности. Далее, воздух не является однородным, так как в результате действия случайных факторов в нем возникают флуктуации колебания плотности. Если пойти ещё глубже, нужно учесть, что по закону всемирного тяготения каждое тело действует на каждое другое тело. Отсюда следует, что даже маятник настенных часов изменяет своим движением траекторию камня.

Короче говоря, если мы всерьез захотим точно исследовать поведение какого-либо предмета, то нам предварительно придется узнать местонахождение и скорость всех остальных предметов Вселенной. А это, разумеется. невозможно.

Чтобы описать явление, необходимо выявить самые существенные его свойства, закономерности, внутренние связи, роль отдельных характеристик явления. Выделив наиболее важные факторы, можно пренебречь менее существенными.

Наиболее эффективно математическую модель можно реализовать на компьютере в виде алгоритмической модели — так называемого "вычислительного эксперимента" (см. [1], параграф 26).

Конечно, результаты вычислительного эксперимента могут оказаться и не соответствующими действительности, если в модели не будут учтены какие-то важные стороны действительности.

Итак, создавая математическую модель для решения задачи, нужно:

1. выделить предположения, на которых будет основываться математическая модель;
2. определить, что считать исходными данными и результатами;
3. записать математические соотношения, связывающие результаты с исходными данными.

При построении математических моделей далеко не всегда удается найти формулы, явно выражающие искомые величины через данные. В таких случаях используются математические методы, позволяющие дать ответы той или иной степени точности.

Существует не только математическое моделирование какого-либо явления, но и визуально-натурное моделирование, которое обеспечивается за счет отображения этих явлений средствами машинной графики, т.е. перед исследователем демонстрируется своеобразный "компьютерный мультфильм", снимаемый в реальном масштабе времени. Наглядность здесь очень высока.

Какие основные этапы содержит процесс разработки программ?

Наличие ошибок в только что разработанной программе это вполне нормальное закономерное явление. Практически невозможно составить реальную (достаточно сложную) программу без ошибок.

Нельзя делать вывод, что программа правильна, лишь на том основании, что она не отвергнута машиной и выдала результаты.

Ведь все, что достигнуто в данном случае, это получение каких-то результатов, не обязательно правильных. В программе при этом может оставаться большое количество логических ошибок.

Как проконтролировать текст программы до выхода на компьютер?

Текст программы можно проконтролировать за столом с помощью просмотра, проверки и прокрутки.

- Просмотр. Текст программы просматривается на предмет обнаружения описок и расхождений с алгоритмом. Нужно просмотреть организацию всех циклов, чтобы убедиться в правильности операторов, задающих кратности циклов. Полезно посмотреть еще раз условия в условных операторах, аргументы в обращениях к подпрограммам и т.п.
- Проверка. При проверке программы программист по тексту программы мысленно старается восстановить тот вычислительный процесс, который определяет программа, после чего сверяет его с требуемым процессом. На время проверки нужно "забыть", что должна делать программа , и "узнавать" об этом по ходу её проверки. Только после окончания проверки программы можно "вспомнить" о том, что она должна делать и сравнить реальные действия программы с требуемыми.
- Прокрутка. Основой прокрутки является имитация программистом за столом выполнения программы на машине. Для выполнения прокрутки приходится задаваться какими-то исходными данными и производить над ними необходимые вычисления. Прокрутка — трудоемкий процесс, поэтому ее следует применять лишь для контроля логически сложных участков программ. Исходные данные должны выбираться такими, чтобы в прокрутку вовлекалось большинство ветвей программы.

Для чего нужны отладка и тестирование?

Отладка программы — это процесс поиска и устранения ошибок в программе, производимый по результатам её прогона на компьютере.

Тестирование — это испытание, проверка правильности работы программы в целом, либо её составных частей.

Отладка и тестирование (англ. test — испытание) — это два четко различимых и непохожих друг на друга этапа:

- при отладке происходит локализация и устранение синтаксических ошибок и явных ошибок кодирования;
- в процессе же тестирования проверяется работоспособность программы, не содержащей явных ошибок.

Тестирование устанавливает факт наличия ошибок, а отладка выясняет ее причину.

Английский термин ***debugging*** ("*отладка*") буквально означает "*вылавливание жучков*". Термин появился в 1945 г., когда один из первых компьютеров — "Марк-1" прекратил работу из-за того, что в его электрические цепи попал мотылек и заблокировал своими останками одно из тысяч реле машины.

В чем заключается отладка?

В современных программных системах (Turbo Basic, Turbo Pascal, Turbo C и др.) отладка осуществляется часто с использованием специальных программных средств, называемых отладчиками. Эти средства позволяют исследовать внутреннее поведение программы.

Программа-отладчик обычно обеспечивает следующие возможности:

- пошаговое исполнение программы с остановкой после каждой команды (оператора);
- просмотр текущего значения любой переменной или нахождение значения любого выражения, в том числе, с использованием стандартных функций; при необходимости можно установить новое значение переменной;
- установку в программе "контрольных точек", т.е. точек, в которых программа временно прекращает свое выполнение, так что можно оценить промежуточные результаты, и др.
- При отладке программ важно помнить следующее:
- в начале процесса отладки надо использовать простые тестовые данные;
- возникающие затруднения следует четко разделять и устранять строго поочередно;
- не нужно считать причиной ошибок машину, так как современные машины и трансляторы обладают чрезвычайно высокой надежностью.

Что такое тест и тестирование?

Как бы ни была тщательно отлажена программа, решающим этапом, устанавливающим ее пригодность для работы, является контроль программы по результатам ее выполнения на системе тестов.

Программу условно можно считать правильной, если её запуск для выбранной системы тестовых исходных данных во всех случаях дает правильные результаты.

Но, как справедливо указывал известный теоретик программирования ►Э. Дейкстра, тестирование может показать лишь наличие ошибок, но не их отсутствие. Нередки случаи, когда новые входные данные вызывают "отказ" или получение неверных результатов работы программы, которая считалась полностью отлаженной.

Для реализации метода тестов должны быть изготовлены или заранее известны эталонные результаты.

Вычислять эталонные результаты нужно обязательно *до*, а не *после* получения машинных результатов.

В противном случае имеется опасность невольной подгонки вычисляемых значений под желаемые, полученные ранее на машине.

Какими должны быть тестовые данные?

Тестовые данные должны обеспечить проверку всех возможных условий возникновения ошибок:

- должна быть испытана каждая ветвь алгоритма;
- очередной тестовый прогон должен контролировать нечто такое, что еще не было проверено на предыдущих прогонах;
- первый тест должен быть максимально прост, чтобы проверить, работает ли программа вообще;
- арифметические операции в тестах должны предельно упрощаться для уменьшения объема вычислений;
- количества элементов последовательностей, точность для итерационных вычислений, количество проходов цикла в тестовых примерах должны задаваться из соображений сокращения объема вычислений;
- минимизация вычислений не должна снижать надежности контроля;
- тестирование должно быть целенаправленным и систематизированным, так как случайный выбор исходных данных привел бы к трудностям в определении ручным способом ожидаемых результатов; кроме того, при случайном выборе тестовых данных могут оказаться непроверенными многие ситуации;
- усложнение тестовых данных должно происходить постепенно.

Пример. Система тестов для задачи нахождения корней квадратного уравнения $ax^2 + bx + c = 0$:

Номер теста	Проверяемый случай	Коэффициенты			Результаты
		a	b	c	
1	d > 0	1	1	-2	x1 = 1, x2 = -2
2	d = 0	1	2	1	Корни равны: x1 = -1, x2 = -1
3	d < 0	2	1	2	Действительных корней нет
4	a = 0, b = 0, c = 0	0	0	0	Все коэффициенты равны нулю. x — любое число
5	a = 0, b = 0, c № 0	0	0	2	Неправильное уравнение
6	a = 0, b № 0	0	2	1	Линейное уравнение; один корень: x = -0.5
7	a № 0, b № 0, c = 0	2	1	0	x1 = 0, x2 = -0.5

Из каких этапов состоит процесс тестирования?

Процесс тестирования можно разделить на три этапа.

- Проверка в нормальных условиях. Предполагает тестирование на основе данных, которые характерны для реальных условий функционирования программы.
- Проверка в экстремальных условиях. Тестовые данные включают граничные значения области изменения входных переменных, которые должны восприниматься программой как правильные данные. Типичными примерами таких значений являются очень маленькие или очень большие числа и отсутствие данных. Еще один тип экстремальных условий — это граничные объемы данных, когда массивы состоят из слишком малого или слишком большого числа элементов.
- Проверка в исключительных ситуациях. Проводится с использованием данных, значения которых лежат за пределами допустимой области изменений.

Известно, что все программы разрабатываются в расчете на обработку какого-то ограниченного набора данных. Поэтому важно получить ответ на следующие вопросы:

? Что произойдет, если программе, не рассчитанной на обработку отрицательных и нулевых значений переменных, в результате какой-либо ошибки придется иметь дело как раз с такими данными?

? Как будет вести себя программа, работающая с массивами, если количество их элементов повысит величину, указанную в объявлении массива?

? Что произойдет, если числа будут слишком малыми или слишком большими?

Наихудшая ситуация складывается тогда, когда программа воспринимает неверные данные как правильные и выдает неверный, но правдоподобный результат.

Программа должна сама отвергать любые данные, которые она не в состоянии обрабатывать правильно.

Каковы характерные ошибки программирования?

Ошибки могут быть допущены на всех этапах решения задачи — от ее постановки до оформления. Разновидности ошибок и соответствующие примеры приведены в таблице:

Вид ошибки	Пример
Неправильная постановка задачи	Правильное решение неверно сформулированной задачи
Неверный алгоритм	Выбор алгоритма, приводящего к неточному или эффективному решению задачи
Ошибка анализа	Неполный учет ситуаций, которые могут возникнуть; логические ошибки
Семантические ошибки	Непонимание порядка выполнения оператора
Синтаксические ошибки	Нарушение правил, определяемых языком программирования
Ошибки при выполнении операций	Слишком большое число, деление на ноль, извлечение квадратного корня из отрицательного числа и т. п.
Ошибки в данных	Неудачное определение возможного диапазона изменения данных
Опечатки	Перепутаны близкие по написанию символы, например, цифра 1 и буквы *I, l*
Ошибки ввода-вывода	Неверное считывание входных данных, неверное задание форматов данных

Является ли отсутствие синтаксических ошибок свидетельством правильности программы?

Обычно синтаксические ошибки выявляются на этапе трансляции. Многие же другие ошибки транслятору выявить невозможно, так как транслятору неизвестны замыслы программиста.

Отсутствие сообщений машины о синтаксических ошибках является необходимым, но не достаточным условием, чтобы считать программу правильной.

Примеры синтаксических ошибок:

- пропуск знака пунктуации;
- несогласованность скобок;
- неправильное формирование оператора;
- неверное образование имен переменных;
- неверное написание служебных слов;
- отсутствие условий окончания цикла;
- отсутствие описания массива и т.п.

Какие ошибки не обнаруживаются транслятором?

Существует множество ошибок, которые транслятор выявить не в состоянии, если используемые в программе операторы сформированы, верно.

Примеры таких ошибок.

Логические ошибки:

- неверное указание ветви алгоритма после проверки некоторого условия;
- неполный учет возможных условий;
- пропуск в программе одного или более блоков алгоритма.

Ошибки в циклах:

- неправильное указание начала цикла;
- неправильное указание условий окончания цикла;
- неправильное указание числа повторений цикла;
- бесконечный цикл.

Ошибки ввода-вывода; ошибки при работе с данными:

- неправильное задание тип данных;
- организация считывания меньшего или большего объёма данных, чем требуется;
- неправильное редактирование данных.

Ошибки в использовании переменных:

- использование переменных без указания их начальных значений;
- ошибочное указание одной переменной вместо другой.

Ошибки при работе с массивами:

- массивы предварительно не обнулены;
- массивы неправильно описаны;
- индексы следуют в неправильном порядке.

Ошибки арифметических операций:

- неверное указание типа переменной (например, целочисленного вместо вещественного);
- неверное определение порядка действий;
- деление на нуль;
- извлечение квадратного корня из отрицательного числа;
- потеря значащих разрядов числа.

Эти ошибки обнаруживаются с помощью тестирования.

В чем заключается сопровождение программы?

Сопровождение программ — это работы, связанные с обслуживанием программ в процессе их эксплуатации.

Многократное использование разработанной программы для решения различных задач заданного класса требует проведения дополнительных работ, связанных с доработками программы для решения конкретных задач, проведения дополнительных тестовых просчетов и т.п.

Программа, предназначенная для длительной эксплуатации, должна иметь соответствующую документацию и инструкцию по её использованию.

Основные термины

Отладка, тестирование, сопровождение программ, ошибки, тестовые данные, математическая модель, результаты тестов.

К о н т р о л ь н ы е в о п р о с ы

1. Какие основные этапы включает в себя решение задач на компьютере?
2. Какие этапы компьютерного решения задач осуществляются без участия компьютера?
3. Что называют математической моделью объекта или явления?
4. Почему невозможно точное исследование поведения объектов или явлений?
5. Какие способы моделирования осуществляются с помощью компьютера?
6. Из каких последовательных действий состоит процесс разработки программы?
7. Доказывает ли получение правдоподобного результата правильность программы?
8. Какие ошибки могут остаться невыявленными, если не провести проверку (просмотр, прокрутку) программы?
9. Чем тестирование программы отличается от её отладки?
10. Каким образом программа-отладчик помогает исследовать поведение программы в процессе её выполнения?
11. Как следует планировать процесс отладки программы?
12. Можно ли с помощью тестирования доказать правильность программы?
13. На какой стадии работы над программой вычисляются эталонные результаты тестов?
14. Назовите основные этапы процесса тестирования.
15. В чём заключается отличие синтаксических ошибок от семантических?
16. О чём свидетельствует отсутствие сообщений машины о синтаксических ошибках?
17. Какие разновидности ошибок транслятор не в состоянии обнаружить?
18. Для чего программам требуется сопровождение?

Тема 13. Алгоритмы. Алгоритмизация. Алгоритмические языки.

План:

1. Что такое алгоритм?

2. Основные свойства алгоритмов
3. Графический способ записи алгоритмов
4. Псевдокод
5. Как записываются алгоритмы на алгоритмическом языке?
6. Базовые алгоритмические структуры
7. Компоненты алгоритмического языка.

Алгоритм — точное и понятное предписание исполнителю совершить последовательность действий, направленных на решение поставленной задачи.

Название "алгоритм" произошло от латинской формы имени среднеазиатского математика аль-Хорезми — Algorithmi. Алгоритм — одно из основных понятий информатики и математики.

Что такое "Исполнитель алгоритма"?

Исполнитель алгоритма — это некоторая абстрактная или реальная (техническая, биологическая или биотехническая) система, способная выполнить действия, предписываемые алгоритмом.

Исполнителя характеризуют:

- среда;
- элементарные действия;
- система команд;
- отказы.

Среда (или обстановка) — это "место обитания" исполнителя. Например, для исполнителя Робота из школьного учебника [1] среда — это бесконечное клеточное поле. Стены и закрашенные клетки тоже часть среды. А их расположение и положение самого Робота задают конкретное состояние среды.

Система команд. Каждый исполнитель может выполнять команды только из некоторого строго заданного списка — системы команд исполнителя. Для каждой команды должны быть заданы условия применимости (в каких состояниях среды может быть выполнена команда) и описаны результаты выполнения команды. Например, команда Робота "вверх" может быть выполнена, если выше Робота нет стены. Ее результат — смещение Робота на одну клетку вверх.

После вызова команды исполнитель совершает соответствующее элементарное действие.

Отказы исполнителя возникают, если команда вызывается при недопустимом для нее состоянии среды.

Обычно исполнитель ничего не знает о цели алгоритма. Он выполняет все полученные команды, не задавая вопросов "почему" и "зачем".

В информатике универсальным исполнителем алгоритмов является компьютер.

Какими свойствами обладают алгоритмы?

Основные свойства алгоритмов следующие:

- Понятность для исполнителя — т.е. исполнитель алгоритма должен знать, как его выполнять.
- Дискретность (прерывность, раздельность) — т.е. алгоритм должен представлять процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов (этапов).
- Определенность — т.е. каждое правило алгоритма должно быть четким, однозначным и не оставлять места для произвола. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.
- Результативность (или конечность). Это свойство состоит в том, что алгоритм должен приводить к решению задачи за конечное число шагов.
- Массовость. Это означает, что алгоритм решения задачи разрабатывается в общем виде, т.е. он должен быть применим для некоторого класса задач, различающихся лишь исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется областью применимости алгоритма.

В какой форме записываются алгоритмы?

На практике наиболее распространены следующие формы представления алгоритмов:

- словесная (записи на естественном языке);
- графическая (изображения из графических символов);
- псевдокоды (полуформализованные описания алгоритмов на условном алгоритмическом языке, включающие в себя как элементы языка программирования, так и фразы естественного языка, общепринятые математические обозначения и др.);
- программная (тексты на языках программирования).

Что такое словесный способ записи алгоритмов?

Словесный способ записи алгоритмов представляет собой описание последовательных этапов обработки данных. Алгоритм задается в произвольном изложении на естественном языке.

Например. Записать алгоритм нахождения наибольшего общего делителя (НОД) двух натуральных чисел.

Алгоритм может быть следующим:

1. задать два числа;
2. если числа равны, то взять любое из них в качестве ответа и остановиться, в противном случае продолжить выполнение алгоритма;
3. определить большее из чисел;
4. заменить большее из чисел разностью большего и меньшего из чисел;
5. повторить алгоритм с шага 2.

Описанный алгоритм применим к любым натуральным числам и должен приводить к решению поставленной задачи. Убедитесь в этом самостоятельно, определив с помощью этого алгоритма наибольший общий делитель чисел 125 и 75.

Словесный способ не имеет широкого распространения по следующим причинам:

- такие описания строго не формализуемы;
- страдают многословностью записей;
- допускают неоднозначность толкования отдельных предписаний.

Что такое графический способ записи алгоритмов?

Графический способ представления алгоритмов является более компактным и наглядным по сравнению со словесным.

При графическом представлении алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий.

Такое графическое представление называется схемой алгоритма или блок-схемой.

В блок-схеме каждому типу действий (вводу исходных данных, вычислению значений выражений, проверке условий, управлению повторением действий, окончанию обработки и т.п.) соответствует геометрическая фигура, представленная в виде блочного символа. Блочные символы соединяются линиями переходов, определяющими очередность выполнения действий.

В таблице приведены наиболее часто употребляемые символы.

Название символа	Обозначение и пример заполнения	Пояснение
Процесс		Вычислительное действие или последовательность действий
Решение		Проверка условий
Модификация		Начало цикла
Предопределенный процесс		Вычисления по подпрограмме, стандартной подпрограмме
Ввод-вывод		Ввод-вывод в общем виде
Пуск-останов		Начало, конец алгоритма, вход и выход в подпрограмму
Документ		Вывод результатов на печать

Блок "процесс" применяется для обозначения действия или последовательности действий, изменяющих значение, форму представления или размещения данных. Для улучшения наглядности

схемы несколько отдельных блоков обработки можно объединять в один блок. Представление отдельных операций достаточно свободно.

Блок "решение" используется для обозначения переходов управления по условию. В каждом блоке "решение" должны быть указаны вопрос, условие или сравнение, которые он определяет.

Блок "модификация" используется для организации циклических конструкций. (Слово модификация означает видоизменение, преобразование). Внутри блока записывается параметр цикла, для которого указываются его начальное значение, граничное условие и шаг изменения значения параметра для каждого повторения.

Блок "предопределенный процесс" используется для указания обращений к вспомогательным алгоритмам, существующим автономно в виде некоторых самостоятельных модулей, и для обращений к библиотечным подпрограммам.

Что такое псевдокод?

Псевдокод представляет собой систему обозначений и правил, предназначенную для единообразной записи алгоритмов.

Он занимает промежуточное место между естественным и формальным языками.

С одной стороны, он близок к обычному естественному языку, поэтому алгоритмы могут на нем записываться и читаться как обычный текст. С другой строны, в псевдокоде используются некоторые формальные конструкции и математическая символика, что приближает запись алгоритма к общепринятой математической записи.

В псевдокоде не приняты строгие синтаксические правила для записи команд, присущие формальным языкам, что облегчает запись алгоритма на стадии его проектирования и дает возможность использовать более широкий набор команд, рассчитанный на абстрактного исполнителя. Однако в псевдокоде обычно имеются некоторые конструкции, присущие формальным языкам, что облегчает переход от записи на псевдокоде к записи алгоритма на формальном языке. В частности, в псевдокоде, так же, как и в формальных языках, есть служебные слова, смысл которых определен раз и навсегда. Они выделяются в печатном тексте жирным шрифтом, а в рукописном тексте подчеркиваются. Единого или формального определения псевдокода не существует, поэтому возможны различные псевдокоды, отличающиеся набором служебных слов и основных (базовых) конструкций.

Примером псевдокода является алгоритмический язык в русской нотации (АЯ), описанный в учебнике А.Г. Кушниренко и др. "Основы информатики и вычислительной техники", 1991. Этот язык в дальнейшем мы будем называть просто "алгоритмический язык".

Как записываются алгоритмы на алгоритмическом языке?

Основные служебные слова

алг (алгоритм)	сим (символьный)	дано	для	да
арг (аргумент)	лит (литерный)	надо	от	нет
рез (результат)	лог (логический)	если	до	при
нач (начало)	таб(таблица)	то	знач	выбор
кон (конец)	нц (начало цикла)	иначе	и	ввод
цел (целый)	кц (конец цикла)	все	или	вывод
вещ (вещественный)	длин (длина)	пока	не	утв

Общий вид алгоритма:
алг название алгоритма (аргументы и результаты)
 дано условия применимости алгоритма
 надо цель выполнения алгоритма
нач описание промежуточных величин
| последовательность команд (тело алгоритма)
кон

Часть алгоритма от слова алг до слова нач называется заголовком, а часть, заключенная между словами нач и кон — телом алгоритма.

В предложении алг после названия алгоритма в круглых скобках указываются характеристики (арг, рез) и тип значения (цел, вещ, сим, лит или лог) всех входных (аргументы) и выходных (результаты) переменных. При описании массивов (таблиц) используется служебное слово таб, дополненное граничными парами по каждому индексу элементов массива.

Примеры предложений алг:

- алг Объем и площадь цилиндра (арг вещ R, H, рез вещ V, S)
- алг Корни КвУр(арг вещ a, b, c, рез вещ x1, x2, рез лит t)
- алг Исключить элемент(арг цел N, арг рез вещ таб A[1:N])
- алг Диагональ(арг цел N, арг цел таб A[1:N,1:N], рез лит Otvet)

Предложения дано и надо не обязательны. В них рекомендуется записывать утверждения, описывающие состояние среды исполнителя алгоритма, например:

1. алг Замена (арг лит Str1, Str2, арг рез лит Text)
2. дано | длины подстрок Str1 и Str2 совпадают
3. надо | всюду в строке Text подстрока Str1 заменена на Str2

1. алг Число максимумов (арг цел N, арг вещ таб A[1:N], рез цел K)
2. дано | N>0
3. надо | K - число максимальных элементов в таблице A

1. алг Сопротивление (арг вещ R1, R2, арг цел N, рез вещ R)
2. дано | N>5, R1>0, R2>0
3. надо | R - сопротивление схемы

Здесь в предложениях дано и надо после знака "|" записаны комментарии. Комментарии можно помещать в конце любой строки. Они не обрабатываются транслятором, но существенно облегчают понимание алгоритма.

Команды АЯ

Оператор присваивания. Служит для вычисления выражений и присваивания их значений переменным. Общий вид: A := B, где знак ":=" означает команду заменить прежнее значение переменной, стоящей в левой части, на вычисленное значение выражения, стоящего в правой части.

Например, a:=(b+c)*sin(Pi/4); i:=i+1.

Для ввода и вывода данных используют команды

- ввод имена переменных
- вывод имена переменных, выражения, тексты.

Для ветвления применяют команды если и выбор, для организации циклов — команды для и пока, описанные в разделе «Что такое базовые алгоритмические структуры?»

Пример записи алгоритма на АЯ

```
алг Сумма квадратов (арг цел n, рез цел S)
дано | n > 0
надо | S = 1*1 + 2*2 + 3*3 + ... + n*n
нач цел i
  ввод n; S:=0
```

```
  нц для i от 1 до n
    S:=S+i*i
  кц
  вывод "S = ", S
кон.
```

Что такое базовые алгоритмические структуры?

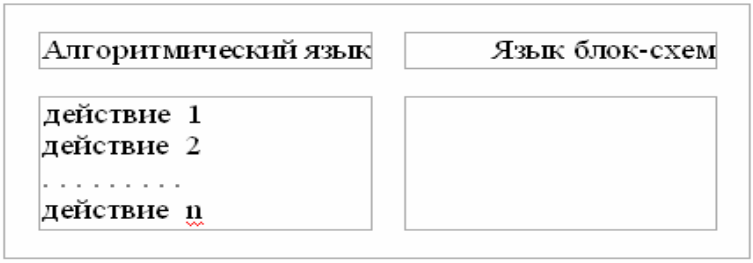
Алгоритмы можно представлять как некоторые структуры, состоящие из отдельных базовых (т.е. основных) элементов. Естественно, что при таком подходе к алгоритмам изучение основных принципов их конструирования должно начинаться с изучения этих базовых элементов. Для их описания будем использовать язык схем алгоритмов и алгоритмический язык.

Логическая структура любого алгоритма может быть представлена комбинацией трех базовых структур:

следование, ветвление, цикл.

Характерной особенностью базовых структур является наличие в них одного входа и одного выхода.

1. Базовая структура следование. Образуется из последовательности действий, следующих одно за другим:



2. Базовая структура ветвление. Обеспечивает в зависимости от результата проверки условия (да или нет) выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу, так что работа алгоритма будет продолжаться независимо от того, какой путь будет выбран.

Структура ветвление существует в четырех основных вариантах:

- если-то;
- если-то-иначе;
- выбор;
- выбор-иначе.

Алгоритмический язык	Язык блок-схем
1. если-то	
если условие то действия все	
2. если-то-иначе	
если условие то действия 1 иначе действия 2 все	
3. выбор	
выбор при условие 1: действия 1 при условие 2: действия 2 . . . . . . при условие N: действия N все	
4. выбор-иначе	
выбор при условие 1: действия 1 при условие 2: действия 2 . . . . . . при условие N: действия N иначе действия N+1 все	

Примеры команды *если*

Алгоритмический язык	Язык блок-схем
если x > 0 то y := sin(x) все	
если a > b то a := 2*a; b := 1 иначе b := 2*b все	
выбор при n = 1: y := sin(x) при n = 2: y := cos(x) при n = 3: y := 0 все	
выбор при a > 5: i := i+1 при a = 0: j := j+1 иначе i := 10; j:=0 все	

3. Базовая структура цикл. Обеспечивает многократное выполнение некоторой совокупности действий, которая называется телом цикла. Основные разновидности циклов представлены в таблице

Алгоритмический язык	Язык блок-схем
Цикл типа *пока*. Предписывает выполнять тело цикла до тех пор, пока выполняется условие, записанное после слова пока.	
нц пока условие тело цикла (последовательность действий) кц	
Цикл типа *для*. Предписывает выполнять тело цикла для всех значений некоторой переменной (параметра цикла) в заданном диапазоне.	
нц для i от i1 до i2 тело цикла (последовательность действий) кц	

Примеры команд *пока* и *для*

Алгоритмический язык	Язык блок-схем
нц пока i <= 5 S := S+A[i] i := i+1 кц	
нц для i от 1 до 5 X[i] := i*i*i Y[i] := X[i]/2 кц	

Какие циклы называют итерационными?

Особенностью итерационного цикла является то,что число повторений операторов тела цикла заранее неизвестно. Для его организации используется цикл типа пока. Выход из итерационного цикла осуществляется в случае выполнения заданного условия.

На каждом шаге вычислений происходит последовательное приближение и проверка условия достижения искомого результата.

Пример. Составить алгоритм вычисления суммы ряда с заданной точностью (для данного знакочередующегося степенного ряда требуемая точность будет достигнута, когда очередное слагаемое станет по абсолютной величине меньше).

Вычисление сумм — типичная циклическая задача. Особенностью же нашей конкретной задачи является то, что число слагаемых (а, следовательно, и число повторений тела цикла) заранее неизвестно. Поэтому выполнение цикла должно завершиться в момент достижения требуемой точности.

При составлении алгоритма нужно учесть, что знаки слагаемых чередуются и степень числа x в числителях слагаемых возрастает.

Решая эту задачу "в лоб" путем вычисления на каждом i-ом шаге частичной суммы **S:=S+(-1)**(i-1)*x**i/i**, мы получим очень неэффективный алгоритм, требующий выполнения большого числа операций. Гораздо лучше организовать вычисления следующим образом: если обозначить числитель какого-либо слагаемого буквой p, то у следующего слагаемого числитель будет равен -p*x (знак минус обеспечивает чередование знаков слагаемых), а само слагаемое m будет равно p/i, где i - номер слагаемого.

Сравните эти два подхода по числу операций.

Алгоритм на АЯ	Блок-схема алгоритма
алг Сумма (арг вещ x, Eps, рез вещ S) дано | 0 < x < 1 надо | S = x - x**2/2 + x**3/3 - ... нач цел i, вещ m, p ввод x, Eps S:=0; i:=1 | начальные значения m:=1; p:=-1 нц пока abs(m) > Eps p:=-p*x | p - числитель | очередного слагаемого m:=p/i | m - очередное слагаемое S:=S+m | S - частичная сумма i:=i+1 | i - номер | очередного слагаемого кц вывод S кон	

Алгоритм, в состав которого входит итерационный цикл, называется итерационным алгоритмом. Итерационные алгоритмы используются при реализации итерационных численных методов.

В итерационных алгоритмах необходимо обеспечить обязательное достижение условия выхода из цикла (сходимость итерационного процесса). В противном случае произойдет зацикливание алгоритма, т.е. не будет выполняться основное свойство алгоритма — результативность.

Что такое вложенные циклы?

Возможны случаи, когда внутри тела цикла необходимо повторять некоторую последовательность операторов, т. е. организовать внутренний цикл. Такая структура получила название цикла в цикле или вложенных циклов. Глубина вложения циклов (то есть количество вложенных друг в друга циклов) может быть различной.

При использовании такой структуры для экономии машинного времени необходимо выносить из внутреннего цикла во внешний все операторы, которые не зависят от параметра внутреннего цикла.

Пример вложенных циклов *для*

Вычислить сумму элементов заданной матрицы A(5,3).

Матрица A	нц для i от 1 до 5 нц для j от 1 до 3 S:=S+A[i,j] кц кц

Пример вложенных циклов *пока*. Вычислить произведение тех элементов заданной матрицы A(10,10), которые расположены на пересечении четных строк и четных столбцов.

```
i:=2; P:=1
нц пока i <= 10
 j:=2
 нц пока j <= 10
  P:=P*A[i,j]
  j:=j+2
 кц
 i:=i+2
кц
```

Чем отличается программный способ записи алгоритмов от других?

При записи алгоритма в словесной форме, в виде блок-схемы или на псевдокоде допускается определенный произвол при изображении команд. Вместе с тем такая запись точна настолько, что позволяет человеку понять суть дела и исполнить алгоритм.

Однако на практике в качестве исполнителей алгоритмов используются специальные автоматы — компьютеры. Поэтому алгоритм, предназначенный для исполнения на компьютере, должен быть записан на "понятном" ему языке. И здесь на первый план выдвигается необходимость точной записи команд, не оставляющей места для произвольного толкования их исполнителем.

Следовательно, язык для записи алгоритмов должен быть формализован. Такой язык принято называть языком программирования, а запись алгоритма на этом языке — программой для компьютера.

Что такое уровень языка программирования?

В настоящее время в мире существует несколько сотен реально используемых языков программирования. Для каждого есть своя область применения.

Любой алгоритм, как мы знаем, есть последовательность предписаний, выполнив которые можно за конечное число шагов перейти от исходных данных к результату. В зависимости от степени детализации предписаний обычно определяется уровень языка программирования — чем меньше детализация, тем выше уровень языка.

По этому критерию можно выделить следующие уровни языков программирования:

- машинные;
- машинно-ориентированные (ассемблеры);
- машинно-независимые (языки высокого уровня).

Машинные языки и машинно-ориентированные языки — это языки низкого уровня, требующие указания мелких деталей процесса обработки данных.

Языки же высокого уровня имитируют естественные языки, используя некоторые слова разговорного языка и общепринятые математические символы. Эти языки более удобны для человека.

Языки высокого уровня делятся на:

- алгоритмические (Basic, Pascal, C и др.), которые предназначены для однозначного описания алгоритмов;
- логические (Prolog, Lisp и др.), которые ориентированы не на разработку алгоритма решения задачи, а на систематическое и формализованное описание задачи с тем, чтобы решение следовало из составленного описания.
- объектно-ориентированные (Object Pascal, C++, Java и др.), в основе которых лежит понятие объекта, сочетающего в себе данные и действия над нами.

Программа на объектно-ориентированном языке, решая некоторую задачу, по сути описывает часть мира, относящуюся к этой задаче. Описание действительности в форме системы взаимодействующих объектов естественнее, чем в форме взаимодействующих процедур.

Какие у машинных языков достоинства и недостатки?

Каждый компьютер имеет свой машинный язык, то есть свою совокупность машинных команд, которая отличается количеством адресов в команде, назначением информации, задаваемой в адресах, набором операций, которые может выполнить машина и др.

При программировании на машинном языке программист может держать под своим контролем каждую команду и каждую ячейку памяти, использовать все возможности имеющихся машинных операций.

Но процесс написания программы на машинном языке очень трудоемкий и утомительный. Программа получается громоздкой, труднообозримой, ее трудно отлаживать, изменять и развивать.

Поэтому в случае, когда нужно иметь эффективную программу, в максимальной степени учитывающую специфику конкретного компьютера, вместо машинных языков используют близкие к ним машинно-ориентированные языки (ассемблеры).

Что такое язык ассемблера?

Язык ассемблера — это система обозначений, используемая для представления в удобочитаемой форме программ, записанных в машинном коде.

Он позволяет программисту пользоваться текстовыми мнемоническими (то есть легко запоминаемыми человеком) кодами, по своему усмотрению присваивать символические имена регистрам компьютера и памяти, а также задавать удобные для себя способы адресации. Кроме того, он позволяет использовать различные системы счисления (например, десятичную или шестнадцатеричную) для представления числовых констант, использовать в программе комментарии и др.

Перевод программы с языка ассемблера на машинный язык осуществляется специальной программой, которая также называется ассемблером и является, по сути, простейшим транслятором.

В чем преимущества алгоритмических языков перед машинными?

Основные преимущества таковы:

- алфавит алгоритмического языка значительно шире алфавита машинного языка, что существенно повы шает наглядность текста программы;
- набор операций, допустимых для использования, не зависит от набора машинных операций, а выбирается из соображений удобства формулирования алгоритмов решения задач определенного класса;
- формат предложений достаточно гибок и удобен для использования, что позволяет с помощью одного пред ложения задать достаточно содержательный этап обра ботки данных;
- требуемые операции задаются с помощью общепринятых математических обозначений;
- данным в алгоритмических языках присваиваются индивидуальные имена, выбираемые программистом;
- в языке может быть предусмотрен значительно более широкий набор типов данных по сравнению с набором машинных типов данных.

Таким образом, алгоритмические языки в значительной мере являются машинно-независимыми. Они облегчают работу программиста и повышают надежность создаваемых программ.

Какие компоненты образуют алгоритмический язык?

Алгоритмический язык (как и любой другой язык) образуют три его составляющие:

алфавит, синтаксис и семантика.

Алфавит — это фиксированный для данного языка набор основных символов, т.е. "букв алфавита", из которых должен состоять любой текст на этом языке — никакие другие символы в тексте не допускаются.

Синтаксис — это правила построения фраз, позволяющие определить, правильно или неправильно написана та или иная фраза. Точнее говоря, синтаксис языка представляет собой набор правил, устанавливающих, какие комбинации символов являются осмысленными предложениями на этом языке.

Семантика определяет смысловое значение предложений языка. Являясь системой правил истолкования отдельных языковых конструкций, семантика устанавливает, какие последовательности действий описываются теми или иными фразами языка и, в конечном итоге, какой алгоритм определен данным текстом на алгоритмическом языке.

Какие понятия используют алгоритмические языки?

Каждое понятие алгоритмического языка подразумевает некоторую синтаксическую единицу (конструкцию) и определяемые ею свойства программных объектов или процесса обработки данных.

Понятие языка определяется во взаимодействии синтаксических и семантических правил. Синтаксические правила показывают, как образуется данное понятие из других понятий и букв алфавита, а семантические правила определяют свойства данного понятия

Основными понятиями в алгоритмических языках обычно являются следующие.

Имена (идентификаторы) — употребляются для обозначения объектов программы (переменных, массивов, функций и др.).

Операции. Типы операций:

- арифметические операции + , - , * , / и др. ;
- логические операции и, или, не;

- операции отношения < , > , <=, >= , = , <> ;
- операция сцепки (иначе, "присоединения", "конкатенации") символьных значений друг с другом с образованием одной длинной строки; изображается знаком "+".

Данные — величины, обрабатываемые программой. Имеется три основных вида данных: константы, переменные и массивы.

- Константы — это данные, которые зафиксированы в тексте программы и не изменяются в процессе ее выполнения.

Примеры констант:

- числовые 7.5, 12;
- логические да (истина), нет (ложь);
- символьные "A", "+";
- литерные "abcde", "информатика", "" (пустая строка).

- Переменные обозначаются именами и могут изменять свои значения в ходе выполнения программы. Переменные бывают целые, вещественные, логические, символьные и литерные.
- Массивы — последовательности однотипных элементов, число которых фиксировано и которым присвоено одно имя. Положение элемента в массиве однозначно определяется его индексами (одним, в случае одномерного массива, или несколькими, если массив многомерный). Иногда массивы называют таблицами.

Выражения — предназначаются для выполнения необходимых вычислений, состоят из констант, переменных, указателей функций (например, exp(x)), объединенных знаками операций.

Выражения записываются в виде линейных последовательностей символов (без подстрочных и надстрочных символов, "многоэтажных" дробей и т.д.), что позволяет вводить их в компьютер, последовательно нажимая на соответствующие клавиши клавиатуры.

Различают выражения арифметические, логические и строковые.

- Арифметические выражения служат для определения одного числового значения. Например, (1+sin(x))/2. Значение этого выражения при x=0 равно 0.5, а при x=p/2 - единице.
- Логические выражения описывают некоторые условия, которые могут удовлетворяться или не удовлетворяться. Таким образом, логическое выражение может принимать только два значения — "истина" или "ложь" (да или нет). Рассмотрим в качестве примера логическое выражение x*x + y*y < r*r , определяющее принадлежность точки с координатами (x,y) внутренней области круга радиусом r с центром в начале координат. При x=1, y=1, r=2 значение этого выражения — "истина", а при x=2, y=2, r=1 — "ложь".
- Значения строковых (литерных) выражений — тексты. В них могут входить литерные константы, литерные переменные и литерные функции, разделенные знаком операции сцепки. Например, A + B означает присоединение строки B к концу строки A. Если A = *"куст "*, а B = *"зеленый"*, то значение выражения A+B есть "*куст зеленый"*.

Операторы (команды). Оператор — это наиболее крупное и содержательное понятие языка: каждый оператор представляет собой законченную фразу языка и определяет некоторый вполне законченный этап обработки данных. В состав операторов входят:

- ключевые слова;
- данные;
- выражения и т.д.

Операторы подразделяются на исполняемые и неисполняемые. Неисполняемые операторы предназначены для описания данных и структуры программы, а исполняемые — для выполнения различных действий (например, оператор присваивания, операторы ввода и вывода, условный оператор, операторы цикла, оператор процедуры и др.).

Что такое стандартная функция?

При решении различных задач с помощью компьютера бывает необходимо вычислить логарифм или модуль числа, синус угла и т.д.

Вычисления часто употребляемых функций осуществляются посредством подпрограмм, называемых стандартными функциями, которые заранее запрограммированы и встроены в транслятор языка.

Таблица стандартных функций алгоритмического языка

Название и математическое обозначение функции		Указатель функции
Абсолютная величина (модуль)	| x |	abs(x)
Корень квадратный		sqrt(x)
Натуральный логарифм	ln x	ln(x)
Десятичный логарифм	lg x	lg(x)
Экспонента (степень числа e " 2.72)	e^x	exp(x)
Знак числа x (-1,если x<0; 0,если x=0; 1,если x>0)	sign x	sign(x)
Целая часть x (т.е. максимальное целое число,не превосходящее x)		int(x)
Минимум из чисел x и y		min(x,y)
Максимум из чисел x и y		max(x,y)
Частное от деления целого x на целое y		div(x,y)
Остаток от деления целого x на целое y		mod(x,y)
Случайное число в диапазоне от 0 до x-1		rnd(x)
Синус (угол в радианах)	sin x	sin(x)
Косинус (угол в радианах)	cos x	cos(x)
Тангенс (угол в радианах)	tg x	tg(x)
Котангенс (угол в радианах)	ctg x	ctg(x)
Арксинус (главное значение в радианах)	arcsin x	arcsin(x)
Арккосинус (главное значение в радианах)	arccos x	arccos(x)
Арктангенс (главное значение в радианах)	arctg x	arctg(x)
Арккотангенс (главное значение в радианах)	arcctg x	arcctg(x)

В качестве аргументов функций можно использовать константы, переменные и выражения. Например:

sin(3.05) min(a, 5)	sin(x) min(a, b)	sin(2*y+t/2) min(a+b, a*b)	sin((exp(x)+1)**2) min(min(a,b),min(c,d))

Каждый язык программирования имеет свой набор стандартных функций.

Как записываются арифметические выражения?

Арифметические выражения записываются по следующим правилам:

- Нельзя опускать знак умножения между сомножителями и ставить рядом два знака операций.
- Индексы элементов массивов записываются в квадратных (АЯ, Pascal) или круглых (Basic) скобках.
- Для обозначения переменных используются буквы латинского алфавита.
- Операции выполняются в порядке старшинства: сначала вычисление функций, затем возведение в степень, потом умножение и деление и в последнюю очередь — сложение и вычитание.
- Операции одного старшинства выполняются слева направо. Например, a/b*c соответствует a/b*c. Однако, в АЯ есть одно исключение из этого правила: операции возведения в степень выполняются справа налево. Так, выражение 2**(3**2) в АЯ вычисляется как 2**(3**2) = 512. В языке QBasic аналогичное выражение 2^3^2 вычисляется как (2^3)^2 = 64. А в языке Pascal вообще не предусмотрена операция возведения в степень, в Pascal x^y записывается как exp(y*ln(x)), а x^y^z как exp(exp(z*ln(y))*ln(x)).

Примеры записи арифметических выражений

Математическая запись	Запись на алгоритмическом языке
	x*y/z
	x/(y*z) или x/y/z
	(a**3+b**3)/(b*c)
	(a[i+1]+b[i-1])/(2*x*y)
	(-b+sqrt(b*b-4*a*c))/(2*a)
(x<0)	sign(x)*abs(x)**(1/5)
	0.49*exp(a*a-b*b)+ln(cos(a*a))**3
	x/(1+x*x/(3+(2*x)**3))

Типичные ошибки в записи выражений:

5x+1	Пропущен знак умножения между 5 и x
a+sinx	Аргумент x функции sin x не заключен в скобки
((a+b)/c**3	Не хватает закрывающей скобки

Как записываются логические выражения?

В записи логических выражений помимо арифметических операций сложения, вычитания, умножения, деления и возведения в степень используются операции отношения < (меньше), <= (меньше или равно), > (больше), >= (больше или равно), = (равно), <> (не равно), а также логические операции и, или, не.

Примеры записи логических выражений, истинных при выполнении указанных условий.

Условие	Запись на алгоритмическом языке
Дробная часть вещественого числа a равна нулю	int(a) = 0
Целое число a — четное	mod(a,2) = 0
Целое число a — нечетное	mod(a,2) = 1
Целое число k кратно семи	mod(a,7) = 0
Каждое из чисел a,b положительно	(a>0) и (b>0)
Только одно из чисел a,b положительно	((a>0) и (b<=0)) или ((a<=0) и (b>0))
Хотя бы одно из чисел a,b,c является отрицательным	(a<0) или (b<0) или (c<0)
Число x удовлетворяет условию a<x<b	(x>a) и (x<b)
Число x имеет значение в промежутке [1, 3]	(x>=1) и (x<=3)

Целые числа a и b имеют одинаковую четность	((mod(a,2)=0) и (mod(b,2)=0) или ((mod(a,2)=1) и (mod(b,2)=1))
Точка с координатами (x,y) лежит в круге радиуса r с центром в точке (a,b)	(x-a)**2+(y-b)**2<r*r
Уравнение ax^2+bx+c=0 не имеет действительных корней	b*b-4*a*c<0
Точка (x,y) принадлежит первому или третьему квадранту	((x>0) и (y>0)) или ((x<0) и (y>0))
Точка (x,y) принадлежит внешности единичного круга с центром в начале координат или его второй четверти	(x*x+y*y>1) или ((x*x+y*y<=1) и (x<0) и (y>0))
Целые числа a и b являются взаимнопротивоположными	a = -b
Целые числа a и b являются взаимообратными	a*b = 1
Число a больше среднего арифметического чисел b,c,d	a>(b+c+d)/3
Число a не меньше среднего геометрического чисел b,c,d	a>=(b+c+d)**(1/3)
Хотя бы одна из логических переменных F1 и F2 имеет значение да	F1 или F2
Обе логические переменые F1 и F2 имеют значение да	F1 и F2
Обе логические переменые F1 и F2 имеют значение нет	не F1 и не F2
Логическая переменная F1 имеет значение да, а логическая переменная F2 имеет значение нет	F1 и не F2
Только одна из логических переменных F1 и F2 имеет значение да	(F1 и не F2) или (F2 и не F1)

Основные термины

Алгоритм, исполнитель алгоритма, место обитания, словесная, программы, псевдокод, базовые алгоритмические структуры, вложенные циклы, язык ассемблера, арифметические выражения, логические выражения.

К о н т р о л ь н ы е в о п р о с ы

1. Что такое «Исполнитель алгоритма»?
2. Какими свойствами обладают алгоритмы?
3. В какой форме записываются алгоритмы?
4. Что такое словесный способ записи алгоритмов?
5. Что такое графический способ записи алгоритмов?
6. Что такое псевдокод?
7. Как записываются алгоритмы на алгоритмическом языке?
8. Что такое базовые алгоритмические структуры?
9. Какие циклы называют итерационными?
10. Что такое вложенные циклы?
11. Чем отличается программный способ записи алгоритмов от других?
12. Какие у машинных языков достоинства и недостатки?
13. Что такое язык ассемблера?
14. Какие компоненты образуют алгоритмический язык?
15. Какие понятия используют алгоритмические языки?
16. Что такое стандартная функция?
17. Как записываются арифметические выражения?
18. Как записываются логические выражения?

Тема 14. Язык программирования Паскаль.
Основы программирования.
План:

1. Системы и языки программирования. Общие сведения о языках программирования.
2. Язык программирования Паскаль. Базовые элементы языка:
- алфавит языка;
- структура программы;
- данные языка;
- встроенные функции.
3. Программирование линейных вычислительных процессов.
Операторы ввода-вывода. Управляющие конструкции языка:
- условный оператор;
- оператор Case;
- оператор перехода;
- ветвление вычислительных процессов.
4. Программирование разветвляющихся вычислительных процессов.
5. Массивы и переменные с индексами.
6. Циклические вычислительные процессы:
- простые циклы;
- оператор цикла с параметром;
- сложные циклы.
7. Программирование циклических вычислительных процессов.
8. Программирование сложных циклических вычислительных процессов.
9. Процедуры и функции, определенные пользователем.
10. Описание процедур и функций.
11. Обращение к процедурам и функциям.
12. Разработка программ с использованием процедур и функций.

СИСТЕМЫ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ.

Составной частью общего (системного) программного обеспечения являются системы программирования с соответствующими алгоритмическими языками.

Системы программирования предназначены для совершенствования процесса разработки и отладки программ. Система программирования включает в свой состав: входной язык системы программирования (называемый также исходным языком); транслятор, обеспечивающий перевод (трансляцию) программы с входного языка системы на внутренний (машинный) язык; библиотеку стандартных, наиболее часто используемых подпрограмм (например, сортировки информации, различного рода встроенных функций и т.п.), подключаемых в процессе подготовки программ к выполнению, а также соответствующую документацию.

Языки программирования, или алгоритмические языки, классифицируются: по степени их зависимости от вычислительной машины; по ориентации на сферу применения; по специфике организационной структуры языковых конструкций и т.п. (рис. 4).

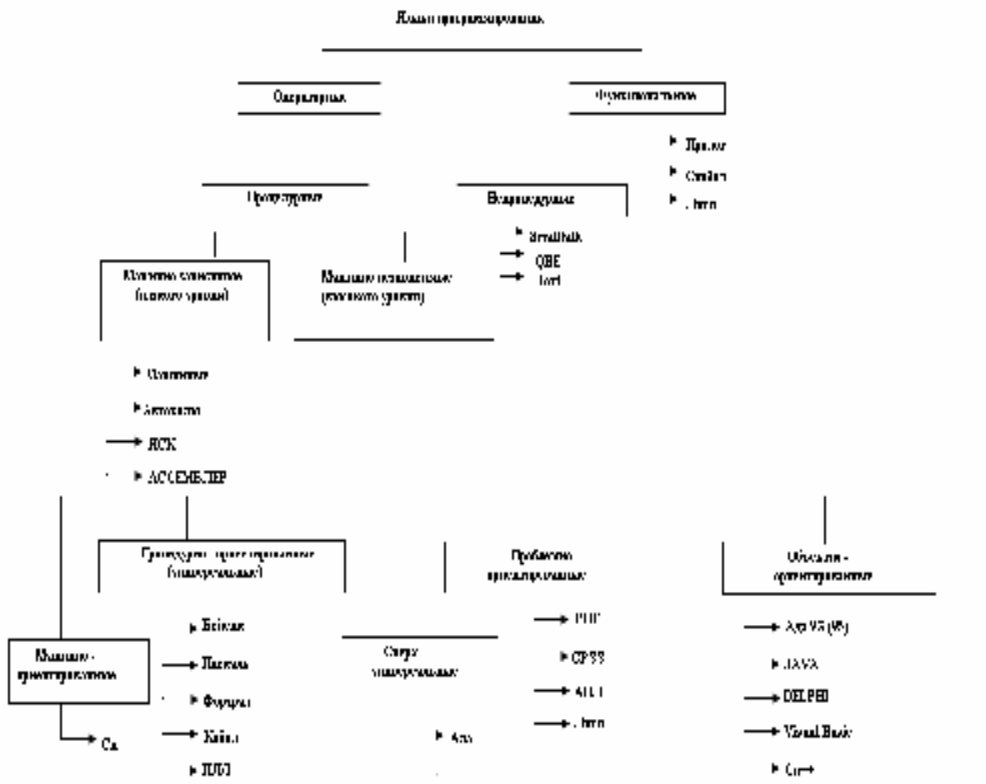


Рис. 4. Классификация языков программирования

С учетом зависимости от ЭВМ языки программирования подразделяются на: машинно-зависимые и машинно-независимые.

Структура и средства машинно-зависимых языков отражают (учитывают) специфику функционирования определенного класса ЭВМ. При программировании задач с помощью таких языков требуется знание не только сущности реализуемого алгоритма решения задачи, но и технических особенностей конкретной ЭВМ и специфики способов написания для нее программ.

К *машинно-зависимым* языкам в первую очередь относятся машинные языки. Машинный язык является внутренним языком ЭВМ и представляет собой систему инструкций и данных, которые не требуют трансляции и могут непосредственно интерпретироваться и исполняться аппаратными средствами ЭВМ. Программирование на этих языках осуществлялось на ЭВМ первого и частично второго поколений.

К машинно-зависимым языкам программирования также относятся машинно-ориентированные языки, основные конструктивные средства которых также позволяют учитывать особенности архитектуры и принципов работы определенной ЭВМ или ряда ЭВМ, т.е. обладают теми же возможностями и требованиями к программистам, что и машинные языки, но в отличие от последних требуют предварительной трансляции на машинный язык программ, составленных с их помощью.

К данному виду языков программирования относятся: автокоды, языки символического кодирования и ассемблеры. В отличие от программирования на машинных языках программирование на машинно-ориентированных языках (ассемблерах) характерно и для современных ПК. Это

объясняется тем, что в языке ассемблера допускается использование средств, присущих языкам высокого уровня.

Использование языка ассемблера, как правило, ограничивается областью системного программирования, т.е. программированием микропроцессоров, разработкой операционных систем или их компонентов, разработкой драйверов - программ обмена информацией между центральными и периферийными устройствами, программ увязки взаимодействия отдельных компонентов прикладных программ и т.д.

Тот факт, что языки данного класса учитывают специфику организации и принципов работы конкретных ЭВМ и допускают при программировании указание конкретных режимов работы физических средств ЭВМ, распределение памяти, явное определение внешних устройств и т.п., относит их к языкам "низкого уровня", или языкам уровня 1:1 (т.е. к языкам, для которых одному оператору входного языка программирования соответствует один оператор машинного языка).

Машинно-независимые языки (или языки высокого уровня) не требуют от пользователя полного знания специфики ЭВМ, на которой реализуется программа решения задачи. Инструментальные средства этих языков программирования позволяют записывать программу в виде, допускающем ее реализацию на ЭВМ с различными типами машинных операций, привязка к которым целиком возлагается на соответствующий транслятор.

Решение задачи на этих языках описывается в наглядном, достаточно легко воспринимаемом виде. Для них характерны: возможность написания выражений, символическая идентификация переменных, вызов функций по именам и т.п. Благодаря этому производительность программиста при составлении исходных программ на языках высокого уровня примерно в 10 -15 раз выше, чем на языке ассемблера. Однако получаемые в результате трансляции машинные программы, как правило, в 2 - 5 раз объемнее по сравнению с такой же программой, но написанной на ассемблере, и работают в 2 - 5 раз медленнее.

Быстрый рост производительности ЭВМ, с одной стороны, и хроническая нехватка программистских кадров, с другой стороны, послужили причиной бурного развития и применения высокоуровневых языков программирования.

Обособленное, промежуточное положение между машинно-независимыми и машинно-зависимыми языками занимает язык Си, создание которого явилось результатом попытки объединения достоинств, присущих языкам обоих классов:

- в плане максимального использования возможностей конкретной вычислительной архитектуры (что присуще языкам низкого уровня), благодаря чему программы на языке Си компактны и работают эффективно;
- в плане максимального использования мощных выразительных возможностей современных языков высокого уровня.

Результат такого компромисса обусловил достаточно сложный синтаксис языка Си.

Язык Си и его модификации в настоящее время используются главным образом для создания системных и прикладных программных продуктов, в которых решающее значение отводится факторам быстродействия и минимизации объемов памяти. На языке Си написано ядро операционной системы Unix, вследствие чего ее легко можно было изменять и модернизировать (а это упрощает процесс ее переноса с одной вычислительной системы на другую, при этом 95% исходного программного текста операционной системы остается неизмененным).

Машинно-независимые языки классифицируются на процедурно-ориентированные и проблемно-ориентированные.

Процедурно-ориентированные (универсальные) языки эффективны для описания алгоритмов решения широкого класса задач. Из языков этого класса наиболее известны: Фортран, Кобол, ПЛ/1, Бейсик, Паскаль, Ада.

Проблемно-ориентированные языки предназначены для описания процессов обработки информации в более узкой, специфической области. Наиболее известными языками этой группы являются: РПГ, Лисп, АПЛ, GPSS.

В последнее время отмечается бурный рост *объектно-ориентированных* языков программирования, т.е. языков, ориентированных на разработку программных приложений для широкого круга разнообразных по сфере приложения задач, имеющих общность в реализуемых компонентах (например, при взаимодействии с базами данных, работе в условиях функционирования

корпоративных сетей организаций или взаимодействии с глобальной сетью Интернет). Объектно-ориентированный подход в программировании позволяет применять одни и те же (типовые) архитектурные и концептуальные решения для быстрого создания эффективных программных приложений.

Основное достоинство алгоритмических языков высокого уровня - возможность описания программ решения задач в форме, максимально удобной для восприятия человеком. Но так как каждое семейство ЭВМ имеет свой собственный, специфический внутренний (машинный) язык и может выполнять лишь те команды, которые записаны на этом языке, то для перевода исходных программ на машинный язык используются специальные программы-трансляторы.

Работа всех трансляторов строится по одному из двух принципов: интерпретация или компиляция.

Интерпретация подразумевает пооператорную трансляцию и последующее выполнение оттранслированного оператора исходной программы. В связи с этим можно отметить два недостатка метода интерпретации: во-первых, интерпретирующая программа должна находиться в памяти ЭВМ в течение всего процесса выполнения исходной программы, т.е. занимать определенный объем памяти; во-вторых, процесс трансляции одного и того же оператора повторяется столько раз, сколько раз должна исполняться эта команда в программе, что резко снижает производительность работы программы.

Несмотря на указанные недостатки, *трансляторы-интерпретаторы* получили достаточное распространение, так как они поддерживают диалоговый режим, что особенно удобно при разработке и отладке исходных программ. Кроме того, интерпретаторы легче разрабатывать, и они обходятся дешевле, чем компиляторы с того же языка.

В случае многократного решения задачи, когда быстродействие работы вычислительной системы имеет существенное значение, целесообразно использовать другой принцип - компиляцию.

При *компиляции* процессы трансляции и выполнения разделены во времени: сначала исходная программа полностью переводится на машинный язык (после чего наличие транслятора в оперативной памяти становится ненужным), а затем оттранслированная программа может многократно исполняться. Следовательно, для одной и той же программы трансляция методом компиляции обеспечивает более высокую производительность вычислительной системы при сокращении требуемой оперативной памяти.

Большая сложность в разработке компилятора по сравнению с интерпретатором с того же самого языка объясняется тем, что компиляция программы включает два действия: анализ, т.е. определение правильности записи исходной программы в соответствии с правилами построения языковых конструкций входного языка, и синтез - генерирование эквивалентной программы в машинных кодах. Трансляция методом компиляции требует неоднократного "просмотра" транслируемой программы, т.е. *трансляторы-компиляторы* являются многопроходными: при первом проходе они проверяют корректность синтаксиса языковых конструкций отдельных операторов независимо друг от друга, при последующем проходе - корректность синтаксических взаимосвязей между операторами и т.д.

Полученная в результате трансляции методом компиляции программа называется *объектным модулем*, который представляет собой эквивалентную программу в машинных кодах, но не "привязанную" к конкретным адресам оперативной памяти. Поэтому перед исполнением объектный модуль должен быть обработан специальной программой операционной системы (редактором связей) и преобразован в *загрузочный модуль*, т.е. программный модуль с относительными адресами.

Загрузочный модуль может иметь простую, оверлейную или динамическую структуру.

Модуль простой структуры состоит из единственного загрузочного модуля, сформированного редактором связей. Этот модуль перед исполнением целиком загружается в оперативную память и включает все необходимые для его работы команды. Модули простой структуры наиболее эффективны с точки зрения производительности, так как в ходе исполнения требуют минимального вмешательства управляющей программы *ОС*.

Если программа функционально достаточно сложна или велика по размерам, то она реализуется в виде *модулей (сегментов) оверлейной структуры* (структуры "с перекрытием"). Загрузочный модуль оверлейной структуры состоит из оверлейных сегментов и содержит информацию, используемую оверлейным супервизором для загрузки отдельных сегментов в основную память. При этом разные сегменты такой программы могут повторно использовать одну и ту же область оперативной памяти.

Оверлейная организация модулей менее эффективна с точки зрения времени их исполнения, так как требует большего вмешательства управляющей программы ОС.

При выполнении модулей простой и оверлейной структуры управляющая программа ОС не осуществляет вызов других загрузочных модулей, и в этом они уступают динамической организации модулей. При выполнении загрузочных *модулей динамической структуры* могут появляться запросы на другие загрузочные модули, и управляющая программа ОС может начать загрузку этих модулей в оперативную память еще до завершения исполнения предыдущего модуля. Благодаря этому вызываемые программные модули могут исполняться как последовательно, так и параллельно, что повышает быстродействие программы.

Наряду с рассмотренными выше трансляторами-интерпретаторами и трансляторами-компиляторами на практике используются также трансляторы *интерпретаторы-компиляторы*, которые объединяют в себе достоинства обоих принципов трансляции: на этапе разработки и отладки программ транслятор работает в режиме интерпретатора, а после завершения процесса отладки исходная программа повторно транслируется в объектный модуль (т.е. уже методом компиляции). Это позволяет значительно упростить и ускорить процесс составления и отладки программ, а за счет последующего получения объектного модуля обеспечить более эффективное исполнение программы.

Учитывая, что эффективность программ, получаемых с помощью высокоуровневых языков программирования, может значительно уступать аналогичным программам, составленным на языках низкого уровня, в ряде случаев используются *оптимизирующие трансляторы (оптимайзеры)*, которые после завершения компиляции осуществляют оптимизацию объектного модуля.

Например, хороший компилятор способен распознать небрежность программиста и исключить из цикла «статические» вычисления. Под «статическими» вычислениями понимается выполнение в цикле определенной операции, результат которой не зависит от итераций цикла. Распознав такую конструкцию, компилятор выводит ее за рамки цикла. Этот тип оптимизации называется перемещением выражений.

Другим более сложным видом оптимизации является устранение неиспользуемого кода. Компилятор находит операторы, которые не выполняются ни при каких условиях, и не включает их в исполняемый код.

В общем случае в зависимости от выигрыша в производительности и временных затрат все виды оптимизации можно разделить на несколько уровней. Первый и второй уровни оптимизации, как правило, повышают быстродействие на 10 - 15% при минимальных затратах. Третий уровень оптимизации позволяет увеличить производительность еще на 5%, однако это обойдется значительно дороже.

Таким образом, выбор типа алгоритмического языка (см. рис. 3) зависит от многих факторов: назначения, удобства написания исходных программ, эффективности получаемых объектных программ и т.п. Для ПК ведущее место в настоящее время занимают языки высокого уровня, например различные версии языков: Бейсик, Паскаль, Фортран, Кобол, АПЛ, Ада, Си и т.д.

Среди них доминирующая роль отводится процедурно-ориентированным языкам, называемым также универсальными (хотя некоторые из них, например Фортран, Кобол, Бейсик и т.п., и разрабатывались с ориентацией на конкретную сферу применения).

Абсолютное большинство языков программирования в настоящее время составляют процедурные языки, с помощью которых программист определяет последовательность реализации событий в объектной программе путем последовательной записи предложений в исходной программе. Иными словами, программирование на этих языках подразумевает необходимость описания не только того, *что* необходимо получить в результате решения задачи, но и того, *как* это необходимо осуществить.

Одним из главных направлений совершенствования языков программирования для задач экономического управления было стремление к разработке таких языков, в которых до минимума сводилась (а в идеале вообще отсутствовала) проблема *как*. Естественно, что такие языки не могут быть процедурными.

Первую попытку создания такого языка предпринял Ломбарди, предложивший и реализовавший в 1963 г. для частного класса задач, связанных с обработкой файлов в сфере управления, язык General business-oriented language baseed on decision expression. В этой же области свои языки предложили фирмы ICL (язык NICOL) и IBM (язык РПГ), в которых многое связанное с процедурой работы

программы носит скорее неявный, чем явный характер. Учитывая, что значительная часть логики программы реализуется автоматически, время создания программ с помощью таких языков и число отладочных проверок значительно сократились.

Таким образом, наряду с универсальными процедурно-ориентированными языками стали создаваться проблемно-ориентированные языки программирования, предназначенные для описания процессов обработки информации в какой-либо узкой (специфической) области, в которых решение задачи в большей степени сосредоточивалось на проблеме, *что* необходимо получить в результате, а проблема, *как* это необходимо сделать, в большей или меньшей степени снималась с программиста. Среди этих языков наиболее известными являются: РПГ - язык для генерации отчетов, Лисп - язык для обработки списков, GPSS - язык для моделирования, АПЛ - язык для статистической обработки массивов.

Актуальности решения проблемы разработки языков программирования, базирующихся на принципах *что* без *как*, способствовал международный конгресс IFIP (International Federation of Information Processing - Международная федерация по обработке информации), прошедший в Стокгольме в 1976 г. под лозунгом *"В 1980 г. программирование без программистов "*. В рамках реализации этой идеи были созданы непроцедурные языки, приближающиеся по своему синтаксису к естественному языку и ориентированные на пользователей - специалистов управления, не являющихся программистами.

Наиболее известными из языков этого типа являются:

- Smalltalking - малый разговорный;
- QBE (Query By Example-программирование на примере);
- Форт, который находит применение при решении сложных задач имитационного моделирования, в системах искусственного интеллекта в графических системах и т.п.

Основной особенностью языка Форт является его открытость, которая позволяет на основе имеющихся определений строить новые функции. При этом программист может вводить новые операции, типы данных или определения. Возможность поддержки средствами Форт многозадачного режима работы придают ему свойства операционной системы.

Особое место среди языков программирования занимают функциональные языки, в частности Пролог (PROLOG –PROgram-ming in LOGic - логическое программирование), предложенный А.Калмероэ в 1978 г., являющийся языком логического программирования, относящимся к языкам пятого поколения. Главное назначение языка - разработка интеллектуальных программ и систем. Пролог - это язык программирования, созданный специально для работы с базами знаний, основанными на фактах и правилах (одного из элементов систем искусственного интеллекта). В языке реализован механизм возврата для выполнения обратной цепочки рассуждений, при котором предполагается, что некоторые выводы или заключения истинны, а затем эти предположения проверяются в базе знаний, содержащей факты и правила логического вывода. Если предположение не подтверждается, выполняется возврат и выдвигается новое предположение.

Языковые средства СУБД предназначены в первую очередь для разработки прикладных программ решения задач экономического управления, информация для которых хранится и поддерживается с помощью баз данных. Синтаксис языка программирования в среде СУБД мало чем отличается от синтаксиса высокоуровневых языков программирования, в связи с чем указанные программно-инструментальные средства ориентированы в основном на профессиональных программистов, хотя наличие развитых средств подсказки и помощи (в виде примеров, демонстрирующих использование отдельных языковых конструкций) значительно облегчает работу достаточно широкого круга пользователей.

Sequel (Structured English QUEry Language) и его усовершенствованный вариант SQL - языки манипулирования данными, основанные на исчислении отношений. Используются в реляционных СУБД в качестве языка запросов к базам данных и языка программирования задач обработки данных.

С развитием компьютерных сетей, увеличением вычислительной мощности компьютеров и их ресурсов возникла потребность в интерпретирующем языке, позволяющем получать многоплатформенную вычислительную среду путем преобразования с его помощью программ, написанных на других языках программирования (при незначительном снижении их производительности). Наиболее близко к реализации подобного языка подошла технология языка Java (а точнее ее часть - байт-код). Именно его разработка и использование составляют принципиальное

отличие, которое выделяет язык Java среди других языков программирования высокого уровня. Объектно-ориентированный язык Java (разработанный на базе языка Си++) предназначен для создания надежных, переносимых, распределенных сетевых программных приложений, работающих в различных многооконных системах в условиях архитектуры клиент-сервер, а также для администраторов сети, использующих Java-приложения для улучшения интерактивных качеств Web-серверов.

Другим объектно-ориентированным языком программирования является язык Delphi. Созданный на базе языка Паскаль специалистами фирмы Borland язык Delphi, обладая мощностью и гибкостью языков Си и Си++, превосходит их по удобству и простоте интерфейса при разработке приложений, обеспечивающих взаимодействие с базами данных и поддержку различного рода работ в рамках корпоративных сетей и сети Интернет.

В последние годы резко расширилась практика программирования в среде электронных таблиц. В основе реализации программирования задач с помощью электронных таблиц (ЭТ) лежит идея компьютеризации работы пользователей с "пустографкой" как специфической формой представления документа, с которым обычно приходится иметь дело специалистам управления. Разработчики электронных таблиц (впоследствии названных табличными процессорами) воплотили эту идею путем предоставления пользователям возможности записи в любую клетку электронной таблицы цифровой, символьной (текстовой) информации либо формулы, обеспечивающей получение искомого результата по различным исходным данным.

Пользователь ЭТ получил удобное инструментальное средство, позволяющее: вводить исходные данные, необходимые для решения задач; указывать формулы получения результатных данных; оформлять решение задачи в виде привычных для него табличных документов. При этом способ описания расчетных формул почти ничем не отличается от принятых правил их представления в математике (за исключением требования линейной структуры записи). Порядок реализации арифметических действий в формулах совпадает с принятым приоритетом выполнения операций в математике, а для изменения такого порядка используется система круглых скобок.

Единственным наследием, доставшимся пользователям ЭТ от программирования на высокоуровневых языках, осталось правило построения логических выражений, реализуемое по стандартной для языков программирования схеме построения условных операторов: IF - THEN - ELSE *(если* условие соблюдается, *то* выполняется *деистие-1, иначе* осуществляется *действие-2).*

Быстрому и широкому распространению ЭТ как инструментальному средству решения экономических задач, помимо простоты и удобства подготовки решения с их помощью задач, способствовали также:

- наличие большого числа встроенных функций (математических, статистических, финансовых и т.п.);
- возможность "проигрывания" различных вариантов решения задач и выбора лучшего из них (за счет быстрого автоматического пересчета конечных результатов при любом изменении исходных данных);
- поддержка ЭТ средствами графической интерпретации, наглядное представление результатной информации.

Кроме того, работа с современными электронными таблицами может быть автоматизирована благодаря использованию командных языков, макросов и т.п.

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ ПАСКАЛЬ

Язык программирования ПАСКАЛЬ является достаточно простым и компактным языком, который широко применяется в мини-ЭВМ и ПК. Конструкции языка позволяют работать не только со стандартными типами данных, но и представляют пользователю возможность выполнять операции над файлами, множествами и записями, использовать динамические структуры данных.

Программа на языке **ПАСКАЛЬ** состоит из заголовка и блока, заканчивающегося точкой. Заголовок программы начинается с ключевого слова **PROGRAM**.

В языке ПАСКАЛЬ используются следующие символы:

1. **Буквы:** 26 прописных латинских букв;
2. **Цифры:** 0, 1, ..., 9;

3. Специальные символы: «: =», « + », «—», «*», «/», «>», «<», « = », «(», «)», «[», «]», «.», «,», «;», «_», «{», «}».

4. Ключевые слова: AND — и, ARRAY —массив, BEGIN — начало, CASE — вариант, CONST —константа, DIV— деление нацело, DO —выполнять, DOWNTO— уменьшать до, END — конец, FILE — файл, FOR — для, FUNKTION — функция, GOTO — перейти на, IF — если, IN — включение, LABEL — метка, MOD — модуль, NIL—отсутствие указателя, NOT — не, OF —из, OR — или, PACKED —упакованный, PROCEDURE — процедура, PROGRAM —программа, RECORD — запись, REPEAT — повторять, SET — множество, THEN —то, TO —до, TYPE —тип, UNTIL — до,VAR—переменная, WHILE—пока,WITH—с.

5. Знаки операций: арифметических: + (сложение), - (вычитание), * (умножение), / (деление), DIV (деление нацело, с отбрасыванием остатка), MOD (нахождение остатка от деления нацело); отношения: > (больше), < (меньше), <= (меньше или равно), >= (больше или равно), = (равно), < > (не равно); логических: NOT (отрицание), OR (логическое сложение), AND (логическое умножение); над множествами: * (пересечение множеств), + (объединение множеств), — (разность множеств), IN (принадлежность множеству).

ПРОСТЕЙШИЕ КОНСТРУКЦИИ ЯЗЫКА

К простейшим конструкциям языка относятся числа, константы, переменные, стандартные функции и выражения. Программы обрабатывают данные, которые могут принимать числовые, логические и литерные значения. В языке используются четыре типа скалярных данных: целые (INTEGER), действительные (REAL), логические (BOOLEAN) и символьные (CHAR).

Числа могут быть **целого** и **действительного** типов.

Числа целого типа не имеют дробной части и записываются последовательностью цифр. Например, 42, — 6, 786, 2000, —2121.

Числа действительного типа записываются с фиксированной точкой в виде целой и дробной частей (например, —4.85, 1.64, —0.25) или с плавающей точкой для значений с десятичным порядком (например, 1.68E — 5, 0.5E5, —12.64E — 4).

Данные логического типа могут иметь значение TRUE или FALSE.

Данные символьного типа принимают значение одной литеры, имеющейся в наборе данной ЭВМ.

Набор литер должен отвечать следующим требованиям: содержать буквы и цифры, упорядоченные в алфавитном порядке и в порядке возрастания; включать символ пробела; каждый символ должен иметь порядковый номер.

В программе целесообразно использовать не сами значения, а их имена (идентификаторы).

Имя — последовательность букв и цифр, начинающихся с буквы. В большинстве ЭВМ трансляторы ограничивают длину имени в восемь символов. Например, **X, У, Z, SUMMA, REZ, REZ2.**

В языке ПАСКАЛЬ имеются стандартные имена, использующиеся для записи стандартных констант (**TRUE, FALSE, MAXINT**), стандартных типов (**INTEGER, REAL, BOOLEAN, CHAR, TEXT**), стандартных файлов (**INPUT, OUTPUT**), стандартных функций (**ABS, SQR, SIN,COS**) и др.

Простейшие конструкции языка записывают, используя числовые, логические, символьные значения и имена данных.

Константа представляется в программе неизменяющимся значением, которое может относиться к следующим типам данных: целые, действительные, логические, символьные или строковые. Например:

1001, -44, 26.85, -0.5E —5, TRUE,’C’,’ 8’,
‘ФАКТОРИАЛ =’ , ‘6.8 +T’

Константы описываются в разделе описания констант, который начинается с ключевого слова CONST. Например:

```
CONST   K=1OO; N=5O;
        PI=3.141592;
        LMP=’P’;
```

Переменная используется для записи значений, изменяющихся в программе, и может принадлежать к различным типам переменных: целые, действительные, логические, символьные.

Различают простые переменные, записывающие своими именами, и переменные с индексом, в записи которых указываются имя массива и индексы этого элемента. Например:

TOR, X, У, Z, VOL, A[1], B[2,1], A[I], B[I,J]

Раздел описания переменных начинается с ключевого слова VAR, после которого перечисляются переменные, массивы и их типы. Например:

```
VAR
        ART, TOP, CP6 : REAL;
        KOL, IMAX     : INTEGER;
        BOK, KI, V    : BOOLEAN;

CONST   HMAX=100; ISTR=20;  ISTB=25;
VAR     COP: ARRAY [1. . . NMAX]  OF REAL;
        NOR: ARRAY [1. . . ISTR, 1. . .ISTB]  OF INTEGER;
```

Стандартные функции используются для вычисления часто встречающихся функций при обработке данных. Обращение к стандартным функциям записывается именем функции, а в скобках указывается аргумент.

Для арифметических функций (абсолютное значение ABS (X) и возведение в квадрат SQR(X) результат имеет тот же тип, что и аргумент, который может быть типа REAL или INTEGER.

Для функций SIN (X), COS (X), EXP (X), LN (X), SQRT (X), ARCTAN (X) аргумент может иметь тип REAL или INTEGER, а результат всегда имеет тип REAL.

Функции выделения целой части числа TRUNC (X) и округления числа ROUND (X) используют аргумент типа REAL, а результат всегда имеет тип INTEGER. Функции упорядочивания типов, использующиеся для нахождения предшествующего PRED (X) и последующего SUCC (X) элементов, могут иметь тип аргумента INTEGER, CHAR или BOOLEAN, а результат имеет тот же тип, что и аргумент. Например, для целочисленного значения *5* результатом функции PRED (5) будет 4, а результатом функции SUCC (5) — 6.

Функции преобразования типов: ORD (X) — определение порядкового номера символа X в наборе символов использует тип аргумента CHAR или BOOLEAN, а результат всегда имеет тип INTEGER.

Функция CHR (I) определяет символ из набора символов по порядковому номеру I и использует аргумент типа INTEGER, а результат имеет тип CHAR; функция ODD (X) определяет нечетность числа. Аргумент имеет тип INTEGER, а результат — тип BOOLEAN. Если аргумент имеет нечетное значение, то результатом является значение TRUE, в противном случае — FALSE.

Выражения определяют последовательность вычисления значения. Выражения могут включать в себя константы, переменные, стандартные функции, которые разделяются скобками и знаками операций.

Порядок вычисления выражения определяется скобками, а при их отсутствии — в соответствии со старшинством операций: операция отрицания (NOT), мультипликативные операции (*, /, DIV, MOD, AND), аддитивные операции (+ , —, OR), операции отношения (<=, <, =, < >, >, >=, IN).

Тип результата выражения зависит от типов операндов, участвующих в операции. Тип результата операций « + », «*», « — » является INTEGER, если оба операнда имеют тип INTEGER, и REAL— в противном случае.

Результатом операции «/» всегда является тип REAL, а результат операций DIV и MOD всегда имеет тип INTEGER, так как аргументы могут быть только типа INTEGER.

Результат выполнения логических операций NOT, OR, AND всегда имеет тип BOOLEAN. Аргументы операций сравнения на равенство и неравенство (=, < >) могут иметь любой тип переменных и констант, а результат всегда имеет тип BOOLEAN.

В операциях сравнения и включения (>, <, >=, <=, IN) аргументы могут быть любого типа, а результат имеет только тип BOOLEAN.

Примеры записи выражений:

```
A*EXP (T*T) - SQRT (X*У*Z)
A*X*X + (4.0*A*B - X*C/2.0)
```

ВВОД И ВЫВОД ДАННЫХ

Для ввода используются операторы:

READ($b_1, b_2, ..., b_n$);

READLN($b_1, b_2, ..., b_n$);

READLN;

где $b_1, b_2, ..., b_n$ - имена переменных, значения которых вводятся.

Оператор READ ($b_1, b_2, ..., b_n$); осуществляет ввод данных из стандартного файла INPUT. Типы вводимых данных должны соответствовать типам переменных в списке оператора ввода.

Оператор READLN ($b_1, b_2, ..., b_n$); осуществляет ввод данных из стандартного файла INPUT и после выбора значения последней переменной обеспечивает переход к началу новой строки файла. При вводе значений целого и действительного типов операторы READ и READLN игнорируют пробелы между значениями.

Оператор READLN; обеспечивает пропуск одной строки в стандартном файле INPUT и переход к началу новой строки.

Ввод переменных логического типа недопустим.

Для вывода информации используются операторы:

WRITE ($b_1, b_2 ... b_n$);

WRITELN ($b_1, b_2 ... b_n$);

WRITELN;

где $b_1, b_2, ..., b_n$ — имена переменных, значения которых выводятся.

Оператор WRITE ($b_1, b_2, ..., b_n$); выполняет вывод значений, соответствующих перечисленным именам в стандартный файл OUTPUT, размещая выводимые значения в одной строке.

Оператор WRITELN ($b_1, b_2, ..., b_n$); выполняет вывод значений в стандартный файл OUTPUT и после вывода последнего значения осуществляет переход к новой строке файла.

Оператор WRITELN; обеспечивает пропуск строки в файле и переход к началу новой строки.

Имена переменных в списке операторов вывода могут принадлежать к целому, действительному, символьному или логическому типам. Ширина поля вывода зависит от типа устройства, используемого в данной ЭВМ. Значения величин действительного типа выводятся в нормализованном виде (мантисса числа с порядком), а целого типа — в обычной форме. Операторы вывода допускают использование указания о ширине поля, отводимого под значение.

Общий вид записи операторов для вывода значений целого типа:

WRITE (b: m);

WRITELN (b: m);

а для вывода действительного типа:

WRITE (b: m: n);

WRITELN (b: m: n);

где b — имя выводимой переменной; m — поле, отводимое под значение и задаваемое константой или выражением целого типа; n — часть поля, отводимого под дробную часть числа.

Например, оператор

WRITE (KOL: 8, NOM: 4);

выделяет на строке под значения переменных KOLи NOM соответственно восемь и четыре позиции, а оператор

WRITELN (SUM: 1O:5);

выделяет под значение SUM десять позиций, из которых пять позиций, отводится под дробную часть числа.

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЛИНЕЙНОЙ СТРУКТУРЫ.

Операторы программы линейной структуры записываются друг за другом в соответствии с порядком, определяемым алгоритмом. Для записи программы такой структуры необходимы операторы присваивания, ввода исходных данных и вывода результатов вычислений.

Программа на языке ПАСКАЛЬ оформляется в виде заголовка программы и блока, заканчивающегося точкой.

Заголовок программы имеет вид

PROGRAM *имя* (INPUT, OUT INPUT);

Имя программы выбирается пользователем.

Блок программы включает в себя разделы: описания меток (LABEL); определения констант (CONST); определения типов (TYPE); описания переменных (VAR); описания функций и процедур (FUNCTION и PROCEDURE).

Раздел операторов включает в себя операторы, которые записываются в соответствии с алгоритмом решения и заключаются в операторные скобки BEGIN и END.

Описания и определения в разделах заканчиваются символом «;». При отсутствии описаний или определений, принадлежащих какому-либо разделу, данный раздел опускается.

Операторы присваивания служат для вычисления значения выражения и присваивания его имени результата.

Общий вид оператора:

v: = a;

где v — имя переменной (результата); a—выражение.

Например:

```
IM: = 0;
P: = 1;
IM: = IM+2;
P: = P*IM;
```

Имя переменной и результат выражения должны принадлежать одному типу. Исключение может составить случай, когда выражение имеет значение целого типа, а имя результата — действительного типа. Например, для описаний и определений вида

```
CONST    N=100;
VAR      T: REAL;
```

справедлив оператор присваивания T:=N;

Пример 1. Составить программу для вычисления высот треугольника со сторонами *a, b, c,* используя формулы

$$h_a = \frac{2}{a}\sqrt{p(p-a)(p-b)(p-c)};$$

$$h_b = \frac{2}{b}\sqrt{p(p-a)(p-b)(p-c)};$$

$$h_c = \frac{2}{c}\sqrt{p(p-a)(p-b)(p-c)};$$

$$p = (a+b+c)/2$$

где ***p=(a+b+c)/2.***

Исходными данными для решения являются значения длин сторон треугольника — a, b, c. Для ввода этих значений следует использовать оператор READ. При этом значения ***a, b, c*** должны быть расположены на одной строке экрана дисплея. В программе используются переменная p для вычисления полупериметра и вспомогательная переменная ***t*** для исключения повторений. Вычисленные значения высот h_a, h_b, h_c необходимо вывести со своими именами, каждую на одной строке.

Программа имеет вид

```
PROGRAMM HTR (INPUT, OUTPUT);
VAR     A, B, C, P, T, HA, HB, HC: REAL;
BEGIN
        READ (A, B, C);
        P: = (A+B+C)/2;
        T: =2*SQRT (P *(P-A) *(P-B) *(P-C));
        HA: =T/A;
        HB: =T/B;
        HC: =T/C;
        WRITELN ('HA=', HA);
        WRITELN ('HB=', HB);
        WRITELN ('HC=', HC)
END.
```

В программе отсутствуют метки, константы, типы, функции и процедуры. Поэтому разделы определения и описания указанных структур отсутствуют.

При записи программ следует использовать имена констант или переменных, наилучшим образом отражающих их назначение. Не рекомендуется одно и то же имя использовать для нескольких последовательно получаемых результатов с целью экономии имен. В языке ПАСКАЛЬ допускается размещать на одной строке несколько операторов. Однако этого следует избегать, так как такое размещение приводит к плохому восприятию программы.

Одним из способов, улучшающих восприятие программы, является применение комментариев. Комментарий представляет собой текст, вставляемый в программу для дополнительных пояснений. Комментарии не воспринимаются и не обрабатываются ЭВМ, поэтому они могут быть записаны на русском языке. Текст комментария заключается между символами «{» и «}» или «(*» и «*)» и может содержать цифры, буквы, ключевые слова и специальные символы.

Например:

(* ПРОГРАММА УПОРЯДОЧИВАНИЯ ЭЛЕМЕНТОВ МАССИВА *)

Комментарии могут использоваться для пояснения назначения программы, констант, переменных, разделов программы или для пояснения используемых методов в алгоритме.

В языке ПАСКАЛЬ символьные переменные и массивы описываются в разделе описания типов. Описание типов имеет вид

TYPE (*имя*) OF CHAR;

где (*имя*) — имя символьной переменной или символьного массива. При этом размеры массива описываются обычным образом (как для скалярных массивов).

Символьным переменным и элементам массивов можно задавать значения с помощью операторов присваивания. Например:

A:='NAME ';

B [1]:='SOBIROV';

Над символьными данными можно выполнять операцию сцепления, которая обозначается символом « + ». Например, оператор C: = A + B [1]; присвоит переменной C значение 'NAME SOBIROV'.

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАЗВЕТВЛЯЮЩЕЙСЯ СТРУКТУРЫ.

Разветвления в программах возникают при необходимости выбора одного из нескольких возможных путей в решении задачи, который может зависеть от исходных данных или промежуточных результатов.

Для организации разветвлений в программах используются операторы перехода, условный и выбора.

Оператор перехода имеет общий вид записи:

GOTO *n*;

где ***n***; — метка оператора.

Метка назначается пользователем и представляет собой число без знака, содержащее не более четырех цифр. Используемые в программе метки должны быть описаны в разделе описания меток. Например:

GOTO 120;
GOTO 250;

Эти операторы осуществляют передачу управления оператором, имеющим соответственно метки 120 и 250.

Использовать оператор GOTO следует только в редких случаях, например, для выхода к концу программы или процедуры в случае неправильного задании данных или выхода из тела цикла. Неправильное использование оператора GOTO усложняет программу за счет многократных переходов вперед или назад по программе, затрудняя ее чтение, отладку и проверку на правильность.

Условный оператор имеет два вида записи:

IF *b* THEN *a*;

IF *b* THEN a_1 ELSE a_2;

где ***b*** — логическое выражение; a_1, a_2, a_3 — операторы простые или составные.

Составной оператор представляет собой последовательность операторов, заключенных в ключевые слова BEGIN и END.

Первый вид записи оператора позволяет организовать выполнение оператора ***a*** при условии, что логическое выражение имеет значение TRUE, а в противном случае оператор ***a*** не выполняется, а выполняется оператор, стоящий за условным. Второй вид записи оператора позволяет производить выполнение оператора a_1 в случае, если логическое выражение имеет значение TRUE, и оператора a_2 — в противном случае.

Примеры записи операторов:

IF N DIV 2*2= N THEN N1: = N ELSE N2: = N*N;

IF X<0.1 THEN V: =EXP (X);

Условный оператор можно расширить за счет вложенности новых условий в операторе. Это приводит к сокращению числа операторов, но одновременно снижает наглядность программы. Новое условие может записываться за ключевыми словами THEN и ELSE. По принятому соглашению в языке ПАСКАЛЬ ключевое слово ELSE всегда относится к ближайшему ему слову IF. Например:

IF b_1 THEN a_1 ELSE IF b_2 THEN a_2 ELSE a_3.

Пример 2. Составить программу для вычисления значения функции

$$Z = x^3/y,$$

где y =sin nx + 0,5.

В программе используются имена N, X для исходных данных, имя Y для промежуточного значения и имя Z для результата.

Для записи программы используются два варианта записи условного оператора.

Вариант 1

```
PROGRAM USL1 (INPUT, OUTPUT);
LABEL    20, 30;
VAR    N: INTEGER;
         X,Y,Z:REAL;
BEGIN
        READ (N, X);
        Y: = SIN (N*X) + 0.5;
        IF Y=0 THEN GOTO   2O;
        Z: =SQR (X) *X/Y;
        WRITE ('Z=', Z);
         GOTO 30;
20: WRITE ('Y=0');
30:
END
```

Вариант 2

```
PROGRAM USL2 (INPUT, OUTPUT);
VAR    N: INTEGER;
         X,Y,Z:REAL;
BEGIN
        READ (N,X);
        Y: = SIN (N*X) + 0.5;
        IF Y=0 THEN WRITE ('Y=0')
              ELSE BEGIN
                           Z: =X*X*X/Y
                           WRITE ('Z=', Z)
                      END
END.
```

Пример 3. Составить программу для вычисления функции

$$z=\begin{cases} \sin x, \text{ если } x \le a \\ \cos x, \text{ если } a < x < b \\ tg\, x, \text{ если } x \ge b \end{cases}$$

Получить разветвление на три ветви можно различными сочетаниями операторов перехода и условного. Однако в языке ПАСКАЛЬ условный оператор можно наращивать за счет использования вложенности условий. С использованием одного условного оператора программа имеет вид

```
PROGRAM USL3 (INPUT, OUTPUT);
VAR   X, A, B, Z: REAL;
BEGIN
      READ (X, A, B);
      IF X<=A THEN Z: =SIN(X)
            ELSE IF X>B THEN Z: =SIN(X)/COS(X)
                  ELSE Z:=COS(X);
      WRITELN (Z)
END.
```

Пример 4. Составить программу для упорядочивания трех чисел ***a, b, c*** по возрастанию таким образом, чтобы имени ***a*** соответствовало наименьшее число, имени ***b*** — среднее, а имени ***c*** — наибольшее.

Программа имеет вид

```
PROGRAM SORT (INPUT, OUTPUT);
LABEL   10, 20, 30;
VAR   A, B, C, H: REAL;
BEGIN
      READ (A, B, C);
      IF A<=B THEN GOTO  1O;
            ELSE BEGIN
                  N: =A;
                  A: =B;
                  B: =N
            END;
10: IF A<=C THEN GOTO 20
            ELSE BEGIN
                  H: =A;
                  A: =C;
                  C: =H
            END;
20: IF B<=C THEN GOTO 30
            ELSE BEGIN
                  H: =B;
                  B: =C;
                  C: =H
            END;
30: WRITELN (A, B, C);
END.
```

Оператор выбора (CASE) обеспечивает выполнение одного оператора (простого или составного) из нескольких возможных. Выбор оператора (последовательности операторов) определяется значением выражения (селектора), которое располагается между ключевыми словами CASE и OF. Значение выражения должно совпадать с константами, стоящими перед операторами. Выражение может принадлежать любому типу, кроме REAL.

Выбор оператора определяется совпадением значения селектора и константы, стоящей перед оператором.

Пример 5. Написать программу для вывода дней недели.

Программа имеет вид:

```
PROGRAM DAYWEEK (INPUT, OUTPUT);
VAR NUMBER: INTEGER;
BEGIN
      READ (NUMBER);
      CASE NUMBER OF
1: WRITELN ('ПОНЕДЕЛЬНИК');
2: WRITELN ('ВТОРНИК');
3: WRITELN ('СРЕДА');
4: WRITELN ('ЧЕТВЕРГ');
5: WRITELN ('ПЯТНИЦА');
6: WRITELN ('СУББОТА');
7: WRITELN ('ВОСКРЕСЕНЬЕ')
   END
END.
```

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ.

Циклическая структура программы позволяет производить многократные вычисления группы операторов при изменении одного или нескольких параметров одновременно. В языке ПАСКАЛЬ имеются операторы цикла FOR, WHILE, REPEAT.

Оператор цикла FOR используется для организации цикла с известным числом повторений.

Общий вид записи:

— при увеличении значения параметра

FOR ***i:*** = m_1 TO m_2 DO ***s***;

при уменьшении значения параметра

FOR ***i:*** = m_1 DOWNTO m_2 DO ***s***;

где ***i***— параметр цикла (не может быть величиной действительного типа); m_1 и m_2— начальное и конечное значения параметра цикла соответственно; ***s*** — тело цикла, состоящее из простого или составного оператора. Значения m_1 и m_2 могут быть записаны константами или выражениями, совпадающими по типу с параметром цикла. Шаг изменения параметра цикла равен 1.

Оператор цикла WHILE используется для организации цикла с неизвестным числом повторений. Общий вид записи оператора:

WHILE ***b*** DO ***s***;

где ***b*** — логическое выражение; ***s*** — тело цикла.

Значения переменных, входящих в условие, должны изменяться в теле цикла, иначе цикл не будет завершен.

Оператор цикла REPEAT используется для организации цикла с неизвестным числом повторений.

Общий вид записи оператора:

REPEAT ***s*** UNTIL ***b***;

где ***s*** — тело цикла; ***b*** — логическое выражение.

В отличие от оператора WHILE в операторе REPEAT проверка условия выполняется в конце оператора, поэтому он обеспечивает хотя бы одно вычисление при значении логического выражения TRUE.

Пример 6. Вычислить и вывести на печать значения функции $y = \frac{a^3}{a^2 + x^2}$ при значениях x, изменяющихся от 0 до 3 с шагом 0,1.

Программа, реализующая этот алгоритм с оператором IF, имеет вид

```
PROGRAM TABULFUN (INPUT, OUTPUT);
```

```
LABEL   50;
VAR   A, X, Y: REAL;
BEGIN
        READ (A);
        X: =0;
50: Y: = (A*A*A)/(SQR (A) + X*X);
        WRITELN (X, Y);
        X: =X+0.1;
        IF X<=3.01 THEN GOTO 5O;
END.
```

Эту же программу можно записать с использованием операторов цикла.

В программе заранее можно определить число повторений и использовать оператор цикла FOR.

```
PROGRAM TABULFUN (INPUT, OUTPUT);
VAR   A, X, Y: REAL;
          I: INTEGER;
BEGIN
        READ (A);
        X: =0;
        FOR I: =1 TO 31 DO
              BEGIN
                    Y: = (A*A*A)/(SQR(A)+X*X);
              WRITELN(X, Y);
                    X: =X+0.1;
 END.
```

Можно разработать универсальную программу для табулирования функции на интервале от $x_{нач}$ до $x_{кон}$ с шагом h_x. В этом случае необходимо предварительно подсчитать число повторений, которое будет использовано в операторе цикла.

Программа имеет вид

```
PROGRAM TABULFUN (INPUT, OUTPUT);
VAR   XN, XK, HX, A, X, Y: REAL;
              I, N          : INTEGER;
BEGIN
        READ (XN, XK, HX, A);
        N: =TRUNC ((XK-XN)/HX) + 1;
        X: =XN;
        FOR I: =1 TO 3N DO
              BEGIN
                    Y: = (A*A*A)/(A*A+X*X);
              WRITELN (X, Y);
                    X: =X+HX
              END
 END.
```

При использовании оператора цикла WHILE в универсальной программе будут отсутствовать описания переменных целого типа N, I и оператор присваивания, вычисляющий число повторений N.

Фрагмент программы с оператором цикла имеет вид

```
WHILE X<=3.01 DO
(тело цикла)
```

Фрагмент программы с оператором цикла REPEAT, аналогичный программе с оператором WHILE , имеет вид

```
REPEAT
(тело цикла)
UNTIL X>3.01
```

Пример 7. Составить программу для вычисления значения членов бесконечного ряда $x, \frac{x^2}{2!}, \cdots, \frac{x^n}{n!}, \cdots$ до члена ряда $\frac{x_i}{i!} \le e$.

Для разработки программы можно использовать операторы цикла WHILE и REPEAT. Тело цикла включает в себя вычисление значения члена ряда, вывод этого значения и вычисление значения ***N***.

Программа с оператором REPEAT имеет вид

```
PROGRAM RYD (INPUT, OUTPUT);
VAR   X, Y, EPS: REAL;
            N: INTEGER;
BEGIN
      READ (X, EPS);
      N: =1;
      Y: =1;
      REPEAT
            Y: =Y*X/N
            WRITELN(Y);
             N: =N+1;
      UNTIL Y<EPS
END.
```

При использовании оператора цикла WHILE тело цикла необходимо заключить в операторные скобки, а оператор цикла записать в виде

WHILE Y>EPS DO

ПРИЕМЫ ПРОГРАММИРОВАНИЯ.

В языке ПАСКАЛЬ имеются большие возможности для реализации типовых алгоритмов вычислений.

Организация цикла с несколькими одновременно изменяющимися параметрами

Пример 8. Составить программу для вычисления функции $Z = \frac{x + y_i}{x - y_i}$, если ***x*** изменяется одновременно с y_i, от начального значения ***a*** с шагом ***h***. Значения y_i являются элементами массива $(y_1, y_2, \cdots, y_{20})$.

В программе количество элементов массива следует задать константой ***N*** в разделе определения констант, а переменные ***z, x, h, a*** и массив ***Y*** описать в разделе описания переменных. Ввод элементов массива можно осуществлять в теле цикла.

Программа имеет вид

```
PROGRAM FUNC (INPUT, OUTPUT);
CONST N=20;
VAR   X, H, A, Z: REAL;
            I: INTEGER;
            Y: ARRAY [1...N] OF REAL;
BEGIN
      READ (H, A);
      X: =A;
      FOR I: =1 TO N DO
            BEGIN
                  READ(Y [I]);
```

```
                    Z :=( X+Y [I])/(X-Y [I]);
                    WRITELN (Z);
                    X: =X+H
                END
        END.
```

Программу для данного условия можно записать с операторами WHILE и REPEAT. В этом случае разделы определения констант и описания переменных остаются без изменения. Фрагмент такой программы, записанный после оператора X:=A; с использованием оператора REPEAT, имеет вид

```
    I: =1;
        REPEAT
            READ (Y [I]);
            Z :=( X+Y [I])/(X-Y [I]);
            WRITE (Z);
            X: =X+H;
            I: =I+1
        UNTIL I>N;
```

Запоминание результатов

Пример 9. Составить программу для вычисления и запоминания функции $z_i = \sqrt{\frac{x_i^2+1}{i}}$, где x_i — элементы массива ($x_1, x_2, \cdots, x_{50}$).

Для вычисления и вывода результатов вычислений удобно использовать цикл FOR.

Программа имеет вид

```
PROGRAM MASZ (INPUT, OUTPUT);
CONST NM=50;
VAR   N, I: INTEGER;
       X, Z: ARRAY [1...NM] OF REAL;
BEGIN
      READ (N);
      FOR I: =1 TO N DO
            BEGIN
                  READ(X [I]);
                  Z [I]:=SQRT (SQR(X [I] + 1)/I))
             END;
      FOR I: =1 TO N DO
             WRITE (Z [I]);
 END.
```

Пример 10. Переписать элементы целочисленного массива $M(M_1, M_2, ..., M_{40})$ кратные пяти, подряд в массив ***M5***. Если такие элементы отсутствуют, то выдать соответствующее сообщение.

Программа имеет вид

```
PROGRAM KR5 (INPUT, OUTPUT);
CONST NM=40;
VAR   K, I, N: INTEGER;
      M, M5: ARRAY [1...NM] OF INTEGER;
BEGIN
   READ (N);
   K: =0;
   FOR I: =1 TO N DO
         BEGIN
              READ (M [I]);
```

```
                IF M [I] DIV 5*5=M [I] THEN
                                        BEGIN
                                            K: =K+1;
                                            M5 [K]:=M [I]
                                        END
            END.
        IF K=0 THEN WRITELN ('ЭЛЕМЕНТОВ КРАТНЫХ 5 НЕТ')
                ELSE
                  FOR I: =1 TO K DO
                        WRITE (M5 [I]: B)
END.
```

Для изменения индекса элементов массива M5 в программе введена переменная с именем K.

Вычисление суммы и произведения

Пример 11. Составить программу для вычисления $z = \sum_{i=1}^{20} \frac{x_i^2}{i}$, где x_i - элемент массива ($x_1, x_2, \ldots, x_{20}$).

Программа имеет вид

```
PROGRAM SUMMA (INPUT, OUTPUT);
CONST NM=20;
VAR   N, I: INTEGER;
        Z: =REAL;
        X: =ARRAY [1...NM] OF REAL;
BEGIN
      READ (N);
      Z: =0;
      FOR I: =1 TO N DO
            BEGIN
            READ(X [I]);
          Z: =Z+SQR(X [I])/I
            END;
      WRITELN ('СУММА Z=', Z)
 END.
```

Пример 12. Составить программу для вычисления произведения ***z*** положительных элементов массива ($x_1, x_2, \ldots, x_{100}$).

Программа имеет вид

```
PROGRAM PROIZ (INPUT, OUTPUT);
CONST NM=100;
VAR   I, N: INTEGER;
        Z: =REAL;
        X: =ARRAY [1...NM] OF REAL;
BEGIN
      READ (N);
      Z: =1;
      FOR I: =1 TO N DO
            BEGIN
            READ(X [I]);
            IF X [I]>0 THEN Z: =Z*X[I]
            END;
      WRITELN ('ПРОИЗВОДЕНИЕ ЭЛЕМЕНТОВ Z=', Z)
 END.
```

Вычисление суммы членов бесконечного ряда с заданной точностью

Пример 13. Составить программу для вычисления суммы членов ряда $z = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - .... + (-1)^n \frac{x^{2n}}{(2n)!} + ... = 1 + \sum_{i=1}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!}$ с точностью до члена ряда, меньшего ε.

Программа имеет вид

```
PROGRAM   RYD (IHPUT, OUTPUT);
VAR     Y, Z, X, EPS: REAL;
          N: INTEGER;
BEGIN
        READ (X.EPS);
        Y: =l;
        Z: =l;
        N: =l;
        REPEAT
                Y: = Y*(- X*X / (2*N + (2*N + (2*N-1))));
                Z: =Z+Y;
                N: =N+1
        UNTILY<EPS;
        WRITE (Z)
END.
```

Вычисление полиномов и производных

Вычисление полинома ***n***-й степени по схеме Горнера сводится к организации цикла с известным числом повторений. Коэффициенты полинома записываются в массив, состоящий из ***n+1*** значения.

Пример 14. Составить программу для вычисления полинома $y = 2x^8 - x^6 + 4x^5 - 5x^2 + 6x + 1$ используя формулу Горнера.

Для ввода коэффициентов полинома необходимо организовать дополнительный цикл.

Программа имеет вид

```
PROGRAM POLNOM (INPUT, OUTPUT);
CONST M=9;
VAR   MD, I: INTEGER;
                X, Y: REAL;
                A: ARRAY [1...M] OF REAL;
BEGIN
        READ (MD);
        FOR l: =l TO MD DO
                READ (A [I]);
        Y: =A [1];
        FOR   l: =2   TO   MD DO
                Y: =Y*X*A [I];
        WRITE(Y)
 END.
```

Нахождение наибольшего и наименьшего значения.

В программе нахождение наибольшего (наименьшего) значения сводится к организации цикла, в котором вычисляется значение функции и сравнивается с предыдущим. В качестве начального значения при нахождении наибольшего берется заведомо меньшее число, например —10^{20}, а при нахождении наименьшего — заведомо большее число, например 10^{20}.

Пример 15. Составить программу для нахождения наименьшего значения функции $y = ae^{-bx}\sin(wx + j)$ при изменении аргумента ***x*** в интервале от 0 до ***c*** с шагом ***h.***

При организации программы с оператором FOR необходимо подсчитать число повторений, зависящее от интервала аргумента и его шага.

При использовании операторов WHILE и REPEAT введение дополнительного параметра не требуется, так как цикл организуется с изменением аргумента.

Программа с оператором FOR имеет вид

```
PROGRAM MIMFUN (INPUT.OUTPUT);
VAR    A, B, C, OMEGA, FI, H, Y, YMIN, X: REAL;
           K, I: INTEGER;
BEGIN
      READ (A, B, C, OMEGA, FI, H);
      YMIN: = 1E19;
      K: =TRUNC(C/H) + l;
      X: =0;
      FOR  I: = 1 TO К DO
           BEGIN
                Y: = A*EXP (-B*X)*SIN (OMEGA*X+FI);
                IF Y<YMIN THEN YMIN: =Y
                X: =X+H
           END;
      WRITE ('YMIN= ', YMIN)
END.
```

В программе с оператором WHILE отсутствует описание переменных ***K*** и ***I***, а собственно цикл имеет вид

```
           WHILE X<=C DO
                BEGIN
                   Y: =A*EXP (-B*X)*SIN (OMEGA*X+FI);
                   IF Y < YMIN THEN YMIN: =Y;
                   X: =X+H
                END;
```

Пример 16. Составить программу для нахождения экстремального значения функции $y = |a|e^{bx+cx^2}$ при изменении аргумента ***x*** от $x_{нач} = 0$ до $x_{кон} = 4$ с шагом ***h***.

В программе для использования оператора цикла FOR после ввода исходных данных определяется число повторений. Выход из цикла осуществляется при условии ***N*Y>N*YM***.

Программа имеет вид

```
PROGRAM EXTREM (INPUT, OUTPUT);
      LABEL 10;
      VAR A, B, C, H, X, XN, XK, Y, YM: REAL;
           I, N, К: INTEGER;
       BEGIN
           READ (A, B, C, H, XN, XK);
           K: =TRUNC ((XK - XN)/H) +1;
           IF C>O THEN N: =l ELSE N: =-l;
           YM: =N*1E19;
           X: =XN;
           FOR I: =1 TO К DO
                BEGIN
                     Y: =ABS (A)*EXP (B*X+C*X*X);
                     IF N*Y<N*YM THEN YM: =Y ELSE GOTO 1O;
                     X: =X+H
                END;
           1O: WRITE (YM, Y)
 END.
```

Нахождение наибольшего и наименьшего в массиве.

Нахождение наибольшего (наименьшего) элемента массива является частным случаем предыдущего приема нахождения наибольшего или наименьшего значения функции. В качестве на-

чального значения наибольшего или наименьшего берется первое значение массива, с которым последовательно сравниваются остальные элементы.

Пример 17. Составить программу для нахождения наибольшего значения элемента массива ($x_1, x_2, \ldots, x_{40}$) и его порядкового номера.

Программа имеет вид

```
PROGRAM MAX(INPUT,OUTPUT);
CONST   N=4O;
VAR XMAX: REAL;
        I, N, D, NMAX: INTEGER;
        X: ARRAY [1..N]  OF  REAL;
BEGIN
        READ (ND);
        FOR   I: =l TO ND DO
                READ(X [I]);
        XMAX: = X [1];
        NMAX: =1;
        FOR   I: =2 TO ND DO
                        IF   X [I]>XMAX THEN BEGIN
                                                XMAX: = X [I];
                                                NMAX: =I;
                                              END;
        WRITELN ('Xmax=', XMAX);
        WRITELN ('Nmax=', NMAX)
 END.
```

Нахождение корней уравнения.

Уточнение значения корня уравнения методами итерации и половинного деления требует организации цикла с неизвестным числом повторений. Для этого в программах используются операторы WHILE и REPEAT, в которых условием окончания цикла является достижение заданной точности.

Пример 18. Составить программу для вычисления наименьшего положительного корня уравнения $x-tgx=0$ с точностью до $\varepsilon=10^{-5}$.

Программа имеет вид

```
PROGRAM   COR(INPUT.OUTPUT);
CONST     PI=3.141592653;
VAR     X0,X1,X2,EPS:REAL;
BEGIN
        READ(X0,EPS);
        REPEAT
                X1:=ARCTAN(X0) + PI;
                X0:=X1;
                X2:=Xl;
        UNTIL    ABS(X2-X1) < EPS;
        WRITE(X1.X2)
END.
```

Пример 19. Составить программу нахождения корня уравнения $x+\sqrt{x}+\sqrt[3]{x}-2{,}5=0$ методом половинного деления на отрезке от 0,4 до 1 с точностью до $\varepsilon=10^{-4}$.

Программа имеет вид

```
        PROGRAM   COR(UNPUT.OUTPUT);
        LABEL    10,20;
        VAR      A, B, DX, FA, FB, FX, X, EPS: REAL;
        BEGIN
                READ (A, B, EPS);
                FA: =A+SQRT (A) + EXP (l/3*LN (A))-2.5;
```

```
        FB: =B + SQRT (B) + EXP (l/3*LN (B))-2.5;
        IF FA*FB>0 THEN BEGIN
                                WRITELN ('КОРНЯ НЕТ');
                                GOTO 1O
                                END;
2O: X: = (A*B)/2;
    DX :=(B-A)/2;
    IF DX<=EPS THEN BEGIN
                                WRITELN ('X =', X);
                                GOTO 1O
                            END
                    ELSE FX: =X+SQRT(X) +EXP (l/3*LN(X))-2.5;
    IF FX*FA>0   THEN BEGIN
                                A: =X;
                                GOTO 2O
                                END
                    ELSE IF   FX*FA<0 THEN BEGIN
                                                    B: =X;
                                                    GOTO   20
                                                END;
10:
 END.
```

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ СО СТРУКТУРОЙ ВЛОЖЕННЫХ ЦИКЛОВ

При записи программ со структурой вложенных циклов необходимо обращать внимание на правильность размещения внешнего и внутреннего циклов. Одни постановки задач допускают смену мест внутреннего и внешнего циклов, а в других постановках такая смена приводит к неправильным результатам.

При записи программ со структурой вложенных циклов зона действия внутреннего цикла должна располагаться в зоне действия охватывающего цикла.

Пример 20. Составить программу для вычисления суммы положительных элементов каждой строки матрицы A (10X8).

В данном примере не безразлично, какой цикл будет внешним. Внешний цикл определяет изменение индексов строк, а внутренний — номера элементов на строке. Перед внутренним циклом необходимо задать начальное значение суммы SUM = 0. Так как числа строк и столбцов определены, можно воспользоваться оператором цикла FOR.

Программа имеет вид

```
PROGRAM SUMSTR (INPUT, OUTPUT);
CONST NM=10;
      MM=8;
VAR SUM: REAL;
    N, M, I, J: INTEGER;
    A: ARRAY [1. .NM, 1. .MM] OF REAL;
    ASTR: ARRAY [1..NM] OF REAL;
BEGIN
    READLN (N, M);
    FOR I: =1 TO N DO
    FOR J: =1 TO M DO
        READ (A [I, J]);
    FOR I: =1 TO N DO
        BEGIN
            SUM: =O;
            FOR J: = 1 TO M DO
```

```
                IF   A [I, J]>0   THEN   SUM: =SUM+A [I, J];
                ASTR [I]:=SUM
            END;
        FOR   I: =l   TO N DO WRITE (ASTR [I]);
    END.
```

Программу можно сократить за счет включения оператора ввода во внутренний цикл. При этом тело внутреннего цикла будет составным оператором:

```
        FOR J: = 1 TO MM DO
            BEGIN
                READ (A [I, J]);
                IF A [I, J] > 0 THEN SUM: = SUM+A [I, J]
            END;
```

Пример 21. Составить программу для упорядочивания элементов массива ($x_1, x_2, \ldots, x_{100}$), расположив их в порядке возрастания в том же массиве.

Упорядочивание элементов массива по возрастанию можно выполнить, используя прием нахождения наименьшего элемента. Решение сводится к многократному нахождению наименьшего элемента массива, начиная с первого и до последнего, затем со второго и т. д. После нахождения наименьшего элемента осуществляется перестановка элементов.

Программа имеет вид

```
PROGRAM SQRT1 (INPUT, OUTPUT);
CONST NMAX=100;
VAR I, J, K, N, ND: INTEJER;
        XMIN: REAL;
        X: ARRAY [1. .NMAX] OF REAL;
BEGIN
        READ (ND);
        FOR I: =1 TO ND DO
            READ(X [I]);
        FOR K: =1 TO ND-1 DO
            BEGIN
                XMIN: =X [K];
                N: =K;
                FOR J: =K+1 TO ND DO
                    IF X [J] <XMIN THEN
                                                BEGIN
                                                    XMIN: =X [J];
                                                    N: =J
                                                END;
                X [N]:=X [K];
                X [K]:=XMIN
            END;
         FOR 1:=1 TO ND DO
            WRITE (X [I])
  END.
```

Во внутреннем цикле находятся наименьший элемент и его порядковый номер, а во внешнем цикле осуществляется перестановка наименьшего и первого из рассматриваемых элементов массива.

Пример 22. Составить программу для упорядочивания элементов массива X ($x_1, x_2, \ldots, x_{100}$), располагая их по возрастанию в том же массиве.

Программа имеет вид

```
PROGRAM SORT2 (INPUT, OUTPUT);
CONST NMAX=1OO;
VAR   I, K, ND: INTEGER;
            Y: REAL;
```

```
            X: ARRAY [1. . NMAX] OF REAL;
BEGIN
      READ (ND);
      FOR   I: =1 TO ND DO
            READ   (X [I]);
      FOR K: =l TO ND-1 DO
            FOR I: =K+l TO ND DO
                  IF X [I] <X [K] THEN BEGIN
                                          Y: =X [K];
                                          X [K]:=X [I];
                                          X [I]:=Y;
                                    END;
      FOR I: =1 TO ND DO
            WRITE (X [I])
  END.
```

В данном примере во внутреннем цикле осуществляется перестановка элементов x_k и x_i в тех случаях, когда выполняется условие $x_k < x_i$.

Пример 23. Составить программу для определения значения аргумента с точностью до $\varepsilon = 0{,}01$, при котором функция $y = ax - \ln x$ достигает минимума, при изменении аргумента от 0,2 до 10.

Для уменьшения числа вычислений вначале шаг изменения аргумента принимается равным 0,2. Начальное и конечное значения аргумента определяются в разделе определения констант. Значение ε задается оператором ввода, а шаг — оператором присваивания.

Программа имеет вид

```
PROGRAM ARG (INPUT, OUTPUT);
LABEL 2O, 3O;
CONST XN=O.2; XK=1O;
VAR A, EPS, X, XMIN, Y, YMIN, HX: REAL;
BEGIN
      HX: =0.2;
      READ (A, EPS);
      X: =XN;
20: YMIN: =1E20;
    REPEAT
            Y: =A*X-LN(X);
            IF Y<YMIN THEN BEGIN
                                YMIN: =Y;
                                YMIN: =X;
                              END;
     X: =X+HX;
     UNTIL X>XK;
     IF HX<=EPS THEN GOTO 3O;
     X: =XMIN-HX;
     HX: =EPS;
     GOTO 20;
30:WRITE (XMIN);
END.
```

Пример 24. Составить программу для вычисления значения определенного интеграла $z = \int_a^b \frac{e^{x-x^2}}{x} dx$ с точностью до ε.

В программе YA и YB — значения подынтегральной функции в точках A и B; S1 и S2 — предыдущая и последующая суммы, вычисленные соответственно с шагом ***h*** и ***h/2***; ***x*** — аргумент.

Программа имеет вид

```
PROGRAM INTEGR (INPUT, OUTPUT);
LABEL 30, 40;
VAR   A,B,HX,Sl,S2,X,YA,YB,Y,EPS:REAL;
      N: INTEGER;
BEGIN
      READ (N, A, B, EPS);
      YA: =EXP (A-A*A)/A;
      YB: =EXP (B-B*B)/B;
      S1:=O;
30:   S2 :=( YA+YB)/2;
      X: =A;
      HX :=( B-A)/N;
      WHILE X<=B DO
      BEGIN
            X: =X+HX;
            Y: =EXP(X-X*X);
            S2:=S2+Y;
      END;
      S2:=S2+HX;
      IF   ABS (S1-S2) <=EPS   THEN BEGIN
                                    WRITELN (S2);
                                    GOTO   40
                                    END
                              ELSE BEGIN
                                    Sl: =S2;
                                    N: =2*N;
                                    GOTO 30;
                                  END;
 4O:END.
```

ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ ПОДПРОГРАММ.

Подпрограммы позволяют разрабатывать программы в виде отдельных частей (модулей), которые могут взаимодействовать между собой. Каждая подпрограмма описывается один раз, при необходимости к ней можно многократно обращаться. В языке ПАСКАЛЬ имеется два вида подпрограмм: функция и процедура. Используемые функции и процедуры должны быть описаны в разделе описания функций и процедур.

Описания подпрограмм включают в себя заголовок подпрограммы, разделы описаний (меток, констант, типов, переменных, а также дополнительных функций и процедур, являющихся локальными по отношению к данной подпрограмме), тело процедуры. В заголовке подпрограммы за ключевым словом (FUNCTION или PROCEDURE) указываются имя подпрограммы, а в скобках — список формальных параметров со своими описаниями. В отличие от процедуры в заголовке функции должен быть определен тип результата, передаваемого функцией.

Описание действий в подпрограмме осуществляется с использованием формальных параметров. Обращение к подпрограмме осуществляется с фактическими параметрами, которые должны соответствовать формальным по числу, типу и месту расположения.

Функции используются для вычисления одного значения. Общий вид записи функции:

FUNCTION ***F*** ($q_1 : T_1; q_2 : T_2; \ldots$):***T***;

(Разделы определений и описаний локальных параметров и подпрограмм)

BEGIN

```
        P1;
        P2;
        .
        .
        .
    END;
```

где ***F*** — функции; q_i — имена формальных параметров; ***T*** — тип имени функции; T_i— типы параметров; P_i— операторы тела функции.

Обращение к функции записывается в правой части оператора присваивания, при этом указывается имя и в скобках — фактические параметры в виде ***F*** (b_1, b_2 ,…), где b_i —фактические параметры.

Пример 25. Составить программу для определения значений ***n!, m!, (n—m)!***, используя функцию при вычислении факториала.

Программа имеет вид

```
PROGRAM FACTL (INPUT, OUTPUT);
    VAR NF, MF, NMF: INTEGER;
    FUNCTION FACT (K: INTEGER): INTEGER;
        VAR PF, I: INTEGER;
        BEGIN
            IF K<0 THEN FACT: =O
                ELSE IF К=О THEN FACT: =1
                                ELSE
                                  BEGIN
                                    PF: =l;
                                    FOR   I: =2 TO К DO
                                    PF: =PF*I;
                                    FACT: =PF
                                   END;
        BEGIN
          READ (N, M);
          NF: =FACT (N);
          MF: =FACT (M);
          NMF: =FACT (N-M);
          WRITE ('NF=', NF,'':4,'MF=', ':4, MF,'NMF=', NMF)
        END.
```

В операторах присваивания программы записаны три обращения к функции: ***FACT (N), FACT (M), FACT (N-M).***

Пример 26. Составить программу для вычисления значения функции $y = ax^2 + bx + d$, где $a=\sum_{i=1}^{n_t} t_i;\ b=\sum_{i=n_t+1}^{100} t_i;\ d=\sum_{i=1}^{20} q_i,$ используя функцию $SUM=\sum_{i=k}^{l} mas_i$

Программа имеет вид

```
PROGRAM FUNY (INPUT, OUTPUT);
    CONST N=1OO;
    TYPE INDEX=1..N;
          VECT=ARRAY [INDEX] OF REAL;
    VAR I, NR, NT, NQ: INTEGER;
        T, Q: VECT;
        Y: REAL;
    FANCTION SUM (MAS: VECT; K: INTEGER; MM: INDEX): REAL;
        VAR J: INTEGER;
            BEGIN
                S: =O;
                FOR J: =1 TO MM DO
```

```
                    S: =S+MAS [J];
                SUM: =S
            END;
    BEGIN
        READ (NR, NT, NQ);
        FOR I: =1 TO NT DO READ (T [I]);
        FOR I: =l TO NQ DO READ (Q [I]);
        Y: =SUM(T, 1, NR)*X*X+SUM(T, NR+l, NT)*X+SUM(Q, 1, NQ);
        WRITE ('Y: =', Y)
    END.
```

Процедуры используются при получении нескольких результатов. Общий вид записи процедуры:

PROCEDURE ***F*** (***q₁ :T₁ ; q₂ :T₂ ; … VAR q₃ :T₃***);

(Разделы определений и описаний локальных параметров и подпрограмм)

BEGIN

P₁;

P₂;

.

.

.

END;

где ***F*** — имя процедуры; q_i — имена формальных параметров; T_i — типы параметров; P_i — операторы тела подпрограммы.

Обращение к процедуре осуществляется оператором процедуры в виде

F (***b₁, b₂,…***);

где ***F*** — имя подпрограммы; где b_i —имена фактических параметров.

Пример 27. Составить программу для вычисления $z=\dfrac{Sh^2a+Sh(a-b)}{Sha+\sqrt{Sh(a^2-b^2)}}$, используя процедуру

Sh $x=\dfrac{e^x-e^{-x}}{2}$.

Программа имеет вид

```
PROGRAM FSN (INPUT, OUTPUT);
VAR A, B, Z, Tl, T2, T3: REAL;
    PROCEDURE SH (X: REAL; VAR R: REAL);
    BEGIN
        R: = (EXP (X)-EXP (-X))/2.0
    END;
BEGIN
    READ (A, B);
    SH (A, T1);
    SH (A-B, T2);
    SH (A*A-B*B, T3);
    Z: = (Tl*Tl-T2) / (T1 + SQRT (T3));
    WRITE (Z = ', Z)
END.
```

Пример 28. Составить программу вычисления функции $z=\dfrac{x^{k_1}x^{k_2}}{s_1+s_2}$, где S_1 **и** S_2 — сумма и количество положительных элементов массива $A(A_1,A_2,....,A_{70})$; S_1 и k_2 — сумма и количество положительных элементов массива $B(B_1,B_2,....,B_{40})$. Для вычисления суммы и количества положительных элементов массива использовать процедуру.

В процедуре с именем SUMKOL вычисляются s — сумма положительных элементов массива MAS и *k* — количество этих элементов.

Программа имеет вид

```
PROGRAM FUNZ (UNPUT, OUTPUT);
CONST   RAZM=1OO;
TYPE INDMAX=l..RAZM;
        VECTOR=ARRAY [INDMAX] OF REAL;
VAR NMA, NMB, I, K1, K2: INTEGER;
      P, Z, S1, S2, X: REAL;
      A, B: VECTOR;
PROCEDURE SUMKOL (MAS: VECTOR; MM: INDMAX; VAR: REAL; K: INTEGER);
VAR J: INTEGER;
BEGIN
      S: =O;
      K: =O;
      FOR J: =l TO MM DO
            IF   MAS [J]>0 THEN BEGIN
                                          S: =S+MAS [J];
                                          K: =K+1;
                                    END;
END;
BEGIN
      READ (NMA);
      FOR I: =I TO NMA DD READ (A [I]);
      SUMKOL (A, NMA, S1, K1);
      READ (NMB);
      FOR I: =l TO NMB DO READ (B [I]);
      SUMKOL (B, NMB, S2, K2);
      READ(X);
      P :=( EXP (K1*P)*EXP (K2*P) / (S1+S2);
      WRITE ('Z=', Z);
 END
 END.
```

Если в процедуре и главной программе используются одни и те же имена параметров (процедура связана с программой посредством глобальных параметров), то процедура может быть без параметров.

Пример 29. Составить программу с процедурой без параметров для вычисления полярных координат $r = \sqrt{x^2 + y^2}$ и $f = arctg(y/x)$ по прямоугольным координатам *x* и *y* (x>0).

Программа имеет вид

```
PROGRAM POLKOR (INPUT, OUTPUT);
VAR X, Y, R, F: REAL;
      N, I: INTEGER;
      PROCEDURE POLAR;
      BEGIN
            R: =SQRT(X*X+Y*Y);
            F: =ARCTAN(Y/X);
      END;
BEGIN
      READ (N);
            FOR I: =1   TO N DO
                  BEGIN
                        READ(X, Y);
                        POLAR;
```

```
                    WRITE(R, F);
                END;
END.
```

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ СМЕШАННОЙ СТРУКТУРЫ С ИСПОЛЬЗОВАНИЕМ СТРУКТУРИРОВАННЫХ ТИПОВ ДАННЫХ

К структурированным типам языка относятся массивы, записи, множества, файлы.

Запись представляет собой структуру, состоящую из фиксированного числа компонентов, называемых полями. Компоненты записи могут принадлежать разным типам. Общий вид записи:

TYPE ***P***= RECORD $C_1:T_1$; $C_2:T_2$… $C_n:T_n$ END;

где ***P*** — имя типа записи; C_i — имя компонента (поля) записи; T_i — имя типа полей.

Например:

```
TYPE    FIO=ARRAY [1. . 15] OF CHAR;
        REZ=ARRAY [1. . 4] OF INTEGER;
        SRBAL=REAL;
        ACTIV=INTEGER;
        STIP=INTEGER;
        STUDENT=RECORD
                    FAM: FIO;
                    EKZ: REZ;
                    SRB: SRBAL;
                    ACT: ACTIV;
                    PRIZ: STIP
                END;
```

В определении базовых типов приняты следующие:

FIO — массив типа CHAR из 15 элементов, содержащий фамилию и инициалы студентов;

REZ — массив целого типа, содержащий результаты четырех экзаменов;

SRBAL — определяет средний балл четырех экзаменов и относится к типу REAL;

ACTIV — определяет участие студентов в общественной работе (1 —участвует, 0 — нет);

STIP — определяет признак назначения стипендии студенту (0 — отсутствие стипендии, 1 — обычная стипендия, 2 — повышенная);

Запись STUDENT содержит значения, соответствующие базовым характеристикам:

FAM — фамилия и инициалы, относящиеся к типу FIO;

EKZ — результаты экзаменов, относящиеся к типу REZ;

ACT— признак участия студента в общественной работе, относящийся к типу ACTIV;

SRB — средний балл студента по результатам сессии, относящийся к типу SRBAL;

PRIZ — признак назначения на стипендию, относящийся к типу STIP.

Обращение к компонентам записи осуществляется выбором имени типа записи и имени поля, разделенных точкой.

Если использовать в программе имя ST как переменную типа GRUP, то:

ST [I].FAM	— переменная типа		FIO
ST [I].FAM [I]	»	»	CHAR
ST [I].EKZ [J]	»	»	INTEGER
ST [I].SRB	»	»	REAL
ST [I].ACT	»	»	INTEGER
ST [I].PRIZ	»	»	INTEGER

Пример 30. Составить программу для вычисления среднего балла и зачисления на стипендию студентов, получивших оценки «4» и «5» или имеющих одну оценку «3», но активно выполняющих общественные поручения. Студент, имеющий все оценки «5», получает повышенную стипендию.

Программа составляется из расчета четырех экзаменов и должна использовать признак 1, если студент активно выполняет общественные поручения, или 0 — в противном случае. Для студентов, имеющих хотя бы одну неудовлетворительную оценку, средний балл не вычислять. Признаками

зачисления на стипендию являются; 0 — отсутствие стипендии; 1 — обычная стипендия; 2 — повышенная стипендия.

Программа с учетом предварительных описаний имеет вид

```
PROGRAM SESEIY (UNPUT, OUTPUT);
LABEL    20;
CONST       NCH=15;
            NK=4;
            NST=25;
 TYPE   FIO=ARRAY [1..NCH]  OF   CHAR;
        REZ=ARRAY [1..NK]   OF   INTEGER;
        SRBAL: REAL;
        ACTIV=INTEGER;
        STIP=INTEGER;
        STUDENT=RECORD FAM: FIO;
                       EKZ: REZ;
                       SRB: SRBAL;
                       ACT: ACTIV;
                       PRIZ: STIP
                  END;
        GRUP=ARRAY [1..NST]  OF   STUDENT;
 VAR  ST: GRUP;
        SUM: REAL;
        I, J: INTEGER;
BEGIN
        FOR I: =1 TO NST DO
            BEGIN
                FOR J: =1 TO NCH DO
                BEGIN
                    WRITE (‘                ‘, J,’ :’);
                    READ (ST [I].FAM [J])
                END;
FOR J: =l TO NK DO READ (ST [I].EKZ [J]));
                READ (ST [I].ACT)
                END;
        FOR  I: =l  TO  NST  DO
                BEGIN
                  SUM: =O;
                  FOR J: =l TO NK DO
                      IF ST [I].EKZ [J] <3 THEN GOTO 20
                                           ELSE SUM: =SUM+ST [I].EKZ [J];
                 ST [I].SRB: = SUM/NK;
                 IF (SUM=2O) AND (ST [I].ACT =1) THEN ST [I].PRIZ:=2
                                                 ELSE  IF  (SUM>=17)  AND  (SUM<20)
                                                       AND (ST [I].ACT=1)
                                                   THEN ST [I].PRIZ:=1;
                2O: ST [I].SRB:=0;
                    ST [I].PRIZ:=0
                END;
                FOR  I: =l TO NST DO
                      BEGIN
                          FOR J: =l TO NCH DO
                          WRITELN (ST [J].FAM [I]);
                          WR1TELN (‘     ‘:4, ST [I].SRB:4:2,’     ‘:4.ST [I].PRIZ)
                           END;
```

END.

Множества представляет собой ограниченный набор различных элементов базового типа, который может быть скалярным или ограниченным. Общий вид записи множественного типа:

TYPE P = SET OF *T;*

где *P* — имя множественного типа; *T* — тип компонентов.

Тип компонентов должен быть базовым. Множества образуются из элементов и состоят из перечисления элементов множества, заключенных в квадратные скобки. Запись [ ] означает пустое множество. Например:

```
TYPE
    STUD=1..25;
    GRUP = SET OF STUD;
```

или

```
TYPE
    M = SET OF 1..3;
```

Переменная M может принимать одно значение из следующего множества: [1,2,3], [1,2], [2,3], [1,3], [1], [2], [3], [ ].

К объектам множественного типа применимы операции: объединения (+), пересечения (*), вычитания (—), проверка на равенство (=) и неравенство (< >), включение (>= или < =), принадлежность множеству IN; Например, выражение с использованием логических операций вида (CH>='A') AND (CH<='Z') может быть заменено на выражение с использованием операции IN:

CH IN ['A'..'Z']

Файлы представляют собой последовательность произвольного числа компонентов одного типа. Естественный порядок компонентов в файле определяется их последовательностью. Описание файла имеет вид

```
TYPE
    P = FILE OF T;
```

где ***P*** **—** имя файла, ***T*** **—** тип компонентов файла.

Например,

TYPE FL = FILE OF REAL;

С каждым описанием новой файловой переменной автоматически вводится дополнительная переменная типа компонентов файла, которая является буферной переменной и обозначается P↑.

Для данного описания буферной переменной является FL↑ типа REAL.

К файлам применимы следующие стандартные функции и процедуры:

EOF **(*иф*)** — функция ***конец файла*** указывает, находится ли файл с именем ***ИФ*** в состоянии ***конец файла***. Если файл пуст, то функция EOF принимает значение TRUE; в противном случае — значение FALSE;

PUT **(*иф*)** — процедура осуществляет запись значения буферной переменной ***иф***↑ в файл ***иф***. Запись производится в том случае, если значение функции EOF равно TRUE. При этом после записи значение функции EOF остается равным TRUE, а значение буферной переменной не определено;

GET **(*иф*)** — процедура осуществляет передвижение по файлу на один компонент. Буферная переменная ***иф***↑ получает значение этого компонента, если файл не пуст. Если следующее значение компонента не существует, то значение функции EOF принимает значение TRUE, а значение буферной переменной ***иф***↑ не определено. Процедура GET определена в тех случаях, когда перед выполнением следующего вызова функция имеет значение FALSE (файл не пуст);

RESET **(*иф*)** — процедура осуществляет поиск начала файла и подготавливает его к просмотру. Если функция EOF имеет значение FALSE, то буферная переменная ***иф***↑ приобретает значение первого компонента файла. В противном случае функция EOF имеет значение TRUE, а значение буферной переменной не определено;

REWRITE **(*иф*)** — процедура предшествует созданию файла ***иф***. Значение ***иф*** заменяется на пустой файл. Функция EOF **(*иф*)** принимает значение TRUE и можно создавать новый файл.

Все действия по созданию и просмотру файлов могут быть описаны с помощью указанных функций и процедур. Привязка реальных файлов к выполненной программе осуществляется по командам, не входящим в состав программы.

Пример 31. Составить программу для вычисления среднего арифметического положительных значений ***z***, представленных в виде файла FL.

Программа имеет вид

```
PROGRAM   SREDZ (FL, OUTPUT);
  TYPE FL=FILE OF REAL;
  VAR S, SR: REAL;
      N: INTEGER;
  BEGIN
            N: =0;
            S: =0;
            RESET (FL);
            REPEAT
              IF FL@>0 THEN BEGIN
                                  S: =S+FL@;
                                  N: =N+1;
                                  END;
                GET (FL@);
            UNTIL   EOF (FL@);
            SR; =S/N;
            WRITE ('SR=', SR)
  END.
```

Программа с использованием оператора WHILE имеет вид

```
PROGRAM SREDZ (FL, OUTPUT);
    TYPE FL=FILE OF REAL;
     VAR S, SR: REAL;
    N: INTEGER;
  BEGIN
        N: =0;
        S: =0;
        RESET (FL);
        WHILE NOT EOF (FL) DO
              BEGIN
                    IF FL@>0 THEN
                                      BEGIN
                                            S: =S+FL@;
                                            N: = N+1; GET (FL)
                                      END;
                              SR: =S/N;
                              WRITE ('SR=', SR)
    END.
```

Основные термины

Языки программирования, машинно – зависимые, машинно – независимые, универсальные языки, интерпретация, трансляторы – интерпретаторы, компиляция, трансляторы – компиляторы, объектный модуль, загрузочный модуль, конструкция языка, ввод и вывод данных, программирование алгоритмов линейной структуры, программирование алгоритмов разветвляющейся структуры, программирование алгоритмов циклической структуры, приёмы программирования, программирование алгоритмов со структурой вложенных циклов, подпрограммы, программирование алгоритмов смешанной структуры с использованием структурированных типов данных, числа, имя, константа, тип, переменная, выражения, оператор присваивания, оператор перехода, условный оператор, оператор цикла, функция, процедура.

К о н т р о л ь н ы е в о п р о с ы

1. Какими свойствами должны обладать программные модули?
2. В чем преимущества использования программно-инструментальных средств при разработке программ?
3. Что понимается под CASE-технологией?
4. Что характерно для машинно-независимых языков?
5. Каково назначение проблемно-ориентированных языков программирования?
6. В чем заключаются отличительные особенности трансляторов интерпретирующего и компилирующего типов?
7. Расскажите о структуре загрузочного модуля.
8. Какие типы величин используются в языке программирования?
9. Указать диапазон значений величин целого и действительного типов.
10. Какие имена переменных допустимы в программе?
11. Как задать тип переменной в программе?
12. Можно ли в качестве операнда в арифметическом выражении использовать: а) имя массива; б) имя стандартной функции, например SIN (Y); в) имя символьной переменной или переменной логического типа?
13. Как организовать пропуск одной, двух строк при выводе?
14. Перечислить действия, реализуемые при выполнении условного оператора.
15. Какие действия выполняются оператором перехода?
16. Что такое вычислительный процесс разветвляющейся структуры?
17. Как организовать разветвление вычислений: а) на две ветви; б) на три ветви?
18. Зачем необходимо при отладке программы тестировать все ветви алгоритма?
19. Указать назначение и правила организации цикла.
20. Перечислить возможные способы организации цикла с заданным числом повторений в изучаемом языке программирования.
21. Что такое итерационный циклический процесс? Его отличия от цикла с заданным числом повторений.
22. Какие два этапа необходимо выделить при нахождении корней уравнений?
23. Каковы условия сходимости метода итераций?
24. Почему при программировании итерационных процессов не используются индексированные переменные для обозначения последовательных приближений к корню? Сколько соседних приближений одновременно используется в вычислениях?
25. Каково условие выхода из цикла при вычислении значения суммы бесконечного ряда?
26. Какие операторы организуют цикл в программе вычисления суммы членов бесконечного ряда?
27. Зачем используются рекуррентные соотношения для вычисления значений члена ряда?
28. Какие средства языка целесообразно использовать для организации циклов с заданным числом повторений?
29. Указать, какие операторы составляют тело цикла.
30. В чем состоят преимущества использования операторов цикла в программах?
31. Указать особенности программ, использующих массивы.
32. Какие операторы языка можно использовать для описания массивов?
33. В чем состоит особенность организации цикла при обработке массивов?

КРАТКИЙ СЛОВАРЬ ТЕРМИНОВ

Абонент - физическое или юридическое лицо, взаимодействующее с системой либо с сетью.

Абонентская система - система, которая поставляет или потребляет информацию.

Адаптер - устройство, обеспечивающее согласование параметров входных и выходных сигналов в системе.

Алгоритм - точное предписание, определяющее процесс, ведущий от варьируемых начальных данных к искомому результату.

Арифметико-логическое устройство - часть процессора, предназначенная для выполнения арифметических и логических операций над данными.

Архитектура ЭВМ - концепция, определяющая модель, общую организационную структуру, выполняемые функции, взаимосвязь устройств, методы кодирования обрабатываемых данных в ЭВМ.

Архитектура безопасности данных - концептуальные положения, определяющие методы и средства защиты данных.

Архитектура «клиент-сервер» - концепция локальной сети, при которой основная часть ее ресурсов размещена на серверах, обслуживающих своих клиентов.

База данных - совокупность взаимосвязанных, хранящихся вместе данных при минимальной избыточности, допускающей их оптимальное использование для одного или нескольких приложений.

База знаний - организованная по особым принципам совокупность знаний, относящихся к какой-либо предметной области.

Банк данных - информационная система, содержащая комплекс специальных методов и средств для поддержки информационной модели предметной области с целью обеспечения информационных запросов пользователей.

Безопасность данных - концепция защиты данных от случайного или преднамеренного их изменения, уничтожения, разглашения или несанкционированного использования.

Бит - минимальная физическая единица информации, или двоичная единица измерения количества информации.

Бод - единица скорости передачи данных, измеряемая количеством битов в секунду.

Брандмауэр - специальная программа в Интернете, с помощью которой можно отфильтровать некоторые типы информации.

Буфер - запоминающее устройство для временного хранения данных и согласования скоростей взаимодействия устройств с разными возможностями.

Видеоадаптер, или видеоконтроллер, - специальная плата ПК, обеспечивающая формирование изображения на экране монитора от информации, передаваемой процессором.

Внешняя память - память компьютера, непосредственно не доступная процессору.

Данные - материальные объекты произвольной формы, выступающие в качестве средства представления информации.

Диалог-способ взаимодействия между объектами, включая процессы и пользователя, со скоростью, достаточной для поддержания комфортной рабочей обстановки.

Дигитайзер - устройство поточечного координатного ввода графических изображений.

Дискретный сигнал - сигнал, имеющий конечное число значений.

Дисплей - устройство ввода, редактирования и визуального отображения информации на экране.

Домен - выделенное множество объектов.

Драйвер - специальная вспомогательная программа, управляющая внешними устройствами ПК или управляющая выполнением программ.

Знание жесткое - знание, которое может быть выражено в виде строгих математических моделей и категорий естественнонаучных теорий.

Знание мягкое - спектр решений, между которыми приходится делать выбор, когда правила и критерии такого выбора жестко не определены.

Идентификация - процесс отождествления какого-либо объекта с одним из известных.

Инструментальное ПО - средство разработки и развития программного обеспечения.

Интегральная схема - миниатюрное электронное устройство, элементы которого соединены конструктивно и технологически.

Интерфейс - определенная строгими стандартами граница между взаимодействующими объектами (пользователями, устройствами, программами, процессами и пр.).

Интерфейс пользователя - порядок, определяющий процедуры взаимодействия пользователя с системой.

Информатика - научная область, изучающая модели, методы и средства преобразования информации.

Информация - мера устранения неопределенности в отношении исхода того или иного события.

Информационная база - вся совокупность информации реального объекта.

Информационный поток - совокупность информационных массивов конкретной деятельности, имеющая динамический характер.

Информационная сеть - сеть для обработки, хранения и передачи данных.

Клавиатура - устройство ручного ввода информации в **ПК.**

Коаксиальный кабель - кабель, состоящий из изолированных друг от друга внутреннего и внешнего проводников.

Кодирование - процесс представления данных последовательностью символов иной формы или значения.

Команда ЭВМ - инструкция, представленная в специальном формате.

Коммуникационная сеть - сеть, основной задачей которой является передача данных.

Компьютерный вирус - специально написанная, небольшая по размерам программа, вызывающая нарушения нормального выполнения различных программ пользователя, порчу файлов, создающая различные помехи при работе ПК.

Контроллер - специализированное устройство (или плата), управляющее работой некоторого периферийного устройства и обеспечивающее его связь с системной платой.

Компьютер - общее название вычислительной машины, предназначенной для выполнения преобразований над вводимыми и хранимыми в ней данными.

Кракер (cracker) - лицо, изучающее компьютерную систему именно с целью ее последующего взлома. Кракеры чаще всего реализуют свои криминальные наклонности в виде написания вирусных программ.

Криптография - способ преобразования данных с целью сделать их не понятными для непосвященных лиц.

Кэш-память - промежуточное запоминающее устройство, работающее со скоростью, обеспечивающей функционирование процессора без режимов ожидания.

Локальная вычислительная сеть - система взаимодействующих и связанных между собой средствами передачи информации компьютеров, размещенных на ограниченной территории.

Макрокоманда - последовательность команд, выделяемая в виде небольшой программы.

Маршрутизация - процесс определения в коммуникационной сети пути, по которому может происходить передача данных.

Массив - упорядоченное множество однотипных элементов данных.

Меню - список команд или функций, предлагаемых пользователю на выбор.

Микропроцессор-процессор, выполненный в одном или нескольких взаимосвязанных полупроводниковых кристаллах интегральных схем.

Модем - устройство преобразования сигналов при передаче их между удаленными компьютерами.

Модуль - функционально законченная часть программы или конструктивно законченный элемент.

Мультимедиа-технология - технология, позволяющая на компьютере объединять в единый комплекс информацию различного характера (графическую, звуковую, текстовую, видео) и управлять ею в режиме диалога.

Мышь - устройство позиционирования, служащее для указания координат на экране.

Накопитель информации - устройство для долговременного хранения больших объемов информации.

Нейрокомпьютер - вычислительная система, аппаратное и программное обеспечение которой ориентировано на реализацию нейросетевых алгоритмов.

Нуль-модем - специальный кабель для временного соединения двух ПК через порты ввода-вывода.

Одноранговая архитектура сети - концепция архитектуры сети, в которой ее ресурсы рассредоточены среди равноправных абонентов.

Окно - средство фрагментации данных при их представлении и обработке.

Оперативная память - память для хранения команд и данных, необходимых процессору для выполнения им операций.

Операционная система (ОС) - комплекс программ для управления и координации работы всех устройств ПК, управления процессом выполнения прикладных программ и обеспечения диалога с пользователем.

Пакет - блок данных, передаваемый между абонентами на сетевом уровне.

Память - обобщенное название устройств в компьютере, предназначенное для хранения данных.

Папка - средство организации и представления системных ресурсов ПК в операционных системах Windows.

Параллельная обработка - модель выполнения прикладных процессов (программ) одновременно группой процессоров.

Пароль - признак, удостоверяющий полномочия пользователя или программы на использование какого-либо ресурса.

Периферийное устройство ПК - устройство, которое непосредственно не размещено на его системной плате.

Пиктограмма - небольшое графическое изображение объекта или действия в виде условного значка.

Поле - часть записи для размещения определенного типа данных.

Порт -точка доступа к устройству либо к программе.

Прикладная программа - программа, описывающая процесс выполнения определенной задачи.

Принтер - устройство вывода данных на бумагу.

Провайдер - организация (юридическое лицо), обеспечивающая работу узла (сайта) в сети Интернет.

Проводник - специальная программа для управления файловой системой в Windows 95.

Программа - формализованное описание последовательности действий устройств компьютера по реализации той или иной задачи.

Программирование - процесс создания программы для ЭВМ.

Программное средство - формализованное описание процесса, обеспечивающего автоматизацию решения на компьютере задач пользователя как независимо, так и с помощью программно-инструментальных средств.

Программное обеспечение ПК - совокупность программ и необходимой документации, обеспечивающих обработку или передачу данных.

Программно-инструментальное средство - комплекс программных продуктов для автоматизации разработки программного обеспечения.

Протокол - стандарт, определяющий способ преобразования информации для ее передачи по сетям.

Процессор - устройство компьютера, служащее для выполнения команд.

Процессор MISC - процессор, работающий с минимальным набором длинных команд.

Процессор RISC - процессор, работающий с сокращенным набором команд.

Рабочая станция - компьютер в сети, специализированный на решении определенных задач пользователя.

Разделение времени - технология работы ПК, предусматривающая чередование во времени нескольких процессов (программ), выполняемых в одном компьютере.

Регистр - устройство для временного хранения данных ограниченного размера.

Реляционная база данных - база данных, организованная в виде набора отношений ее компонентов.

CASE-технология - совокупность средств системного анализа, проектирования, разработки и сопровождения сложных программных систем, поддерживаемых комплексом взаимосвязанных инструментальных средств автоматизации всех этапов разработки программ.

Сервер - как правило, компьютер высокой производительности, предоставляющий сервис другим компьютерам сети.

Сервис сетевой - служебные программы, функционирующие на сервере в фоновом режиме и обеспечивающие поддержку других приложений локальной сети в любой момент, с высокой скоростью и надежностью, а также с максимальной простотой (максимальным удобством).

Сеть - взаимодействующая совокупность объектов, образуемых устройствами передачи и обработки данных.

Сеть Интернет - глобальная международная ассоциация информационных сетей.

Система - организованное множество, образующее целостное единство, направленное на достижение определенной цели.

Сканер - устройство автоматизированного ввода графической и текстовой информации в компьютер.

Сообщение - набор данных со смысловым содержанием, пригодных для обработки и передачи.

Список - упорядоченная последовательность произвольных элементов данных.

Стриммер - устройство для хранения и воспроизведения больших объемов информации на кассетную магнитную ленту.

Текстовый редактор - обобщенное название комплекса прикладных программ для создания и редактирования текстов, программ и документов.

Телеконференция - метод проведения дискуссий между удаленными группами пользователей в сети Интернет.

Терминал - устройство ввода-вывода данных и команд в компьютер или в сеть.

Технология - совокупность методов обработки, изготовления, изменения состояния, свойств, формы сырья и материалов, включая информацию, в процессе производства конечной продукции.

Транслятор - специальная программа перевода исходной программы на машинный язык компьютера.

Утилита - программа вспомогательного или служебного назначения для ПК.

Управление - функция системы, ориентированная на сохранение ее основного качества в условиях изменения среды либо на выполнение целевой программы обеспечения устойчивости ее функционирования при достижении заданной цели.

Управление - процесс, предполагающий выполнение функций по сбору, передаче, хранению, обработке и анализу информации, необходимых для выработки соответствующих решений.

Файл - поименованная совокупность данных в памяти ПК или на машинном носителе.

Формат - структура информационного объекта.

Форматирование диска ~ процесс записи на него управляющей информации, определяющей точки начала и конца отдельных секторов диска.

Фракер - приверженец электронного журнала "Phrack", публикующего материалы по новинкам операционных систем и сетевых технологий, использующий принципы построения протоколов сетевого обмена для воздействия на компьютерные системы.

Хакер - лицо, проявляющее чрезмерный интерес к устройству сложных компьютерных систем и вследствие этого обладающее большими познаниями, обеспечивающими им «взлом» информационных и компьютерных систем.

Шина - средство для обеспечения взаимодействия близко расположенных объектов или средство, к которому одинаковым образом подключается группа взаимодействующих друг с другом компьютеров или их устройств.

Шрифт — набор форм символов алфавита, служащий для восприятия устройствами компьютера и людьми.

Экономическая информация - информация, отражающая и обслуживающая процессы производства, распределения, обмена и потребления материальных продуктов и благ.

Экспертная система - система, объединяющая возможности компьютера со знаниями и опытом эксперта в такой форме, что она может предложить разумный совет или разумное решение задачи с пояснением хода своих рассуждений в понятной человеку форме.

Электронная почта - средства передачи сообщений по сети без применения бумажного носителя.

Электронная таблица - распространенное название комплекса прикладных программ для обработки таблиц.

Эргономика - наука о человеке в конкретных условиях его деятельности.

Ярлык - в операционной системе Windows определяется как файл, содержащий путь к объекту.
Ячейка - адресуемый элемент однородной структуры, например таблицы.

Литература

1. Айков Д. и др. Компьютерные преступления.-М.: Мир, 1999.
2. Ахметов К., Борзенко А. современный персональный компьютер.-М.: Компьютер пресс, 1995.
3. Васильев А., Андреев А.VBA в Jffice 2000: Учебный курс.-СПб: Питер, 2000.
4. Вычислительные системы, сети и телекоммуникации: Учебник.-2-е изд., перераб. и.доп /Под ред. А.Б.Пятибратова.-М.: Финансы и статистика, 2001.
5. Гирасименко В.А. Защита информации в автоматизированных системах абработки данных. В 2-х кн. –М.: Энергоатомиздат, 1994.
6. Диго С.М. Проектирование и использование баз данных. –М.: Финансы и статистика, 1995.
7. Зубанов Ф. Mikrosoft Windovs NT –выбор профи.-М.:Русская редакция,1997.
8. Информатика: Учебник / Под ред. Н.В.Макаровой. –3-е изд.,перераб. – М.: Финансы и статистика, 2001.
9. Информатика: Практикум по технологии работы на компьютере / под ред. Н.В. Макаровой.-3-е изд., перераб. – М.: Финансы и статистика, 2000.
10. Кент П. Интернет: Пер. с англ. – М.: ЮНИТИ, 1996.
11. Компьютерные системы и сети: Учеб. Пособ./ Под ред. В.П.Косареваи Л.В Еремина.-М.: Финансы истатистика, 1999.
12. Левин А. Самоучитель работы на компьютере.-М.: 1996.
13. Левин Дж.Р., Бароди К., Левинг Янг М. Internet для «чайников»: Пер.с англ. –3-е езд.-Киев: Диалектика, 1995.
14. Локальные вычеслительные сети. В 3-х кн./ Под ред. С.В.Назарова.-М.: Финансы и статистика, 1995.
15. Мельников В.В. Защита информации в компьютерных системах.-М.: Финансы и статистика, 1997.
16. Назаров С.В. Администририование локальных сетей Windows NT –М.: Финансы и статистика, 1999.
17. Назаров С.В., Мельников П.П. Программирование на MS visual basic.-М.: Финансы и статистика, 2001.
18. Нанс Б. Компьютерные сети: пер. с англ.- М,: Восточная книжная компания, 1996.
19. Ойхман Е.Г., Попов Э.В. Реинжиниринг бизнеса: Реинженириинг организации и информационный технологии.-М.: Финансы и статистика, 1997.
20. Пакеты программ офисного назначения/ Под ред. С.В.Назарова. .-М.: Финансы и статистика, 1997.
21. Практикум по экономической информатике: Учеб. Пособ. –Ч. 1/ Под ред. Е.Л Шуремова, Н.А Тимаковой, Е.А Мамонтовой.-М.: Перспектива, 2000.
22. Росе Дж. Азбука Mikrosoft Internet Explorer 3: Пер. С англ.-М.: Мир, 1997.
23. Рош У. Библия по модернизации персонального компьютера.-Минск, ИПП «Тивали-Стиль» ,1995.
24. Сигалов А. Жёлтые страницы Internet-1997. Русские ресурсы.-СПб.: Питер, 1997.
25. Тайли Э. Безопасность персонального компьютера: Пер. с англ.-Минск:ООО «Попурри» ,1997.
26. Уолл Девид и др. Использование World Wide Web: Пер. с англ.-Киев: Диалектика,1997.
27. Фигурнов В.Э. IBM PC для пользователя.-М.: Инфра –М,1999.
28. Фридман А.Л. Основы обектно –ориентированный разработки программых систем.- .-М.: Финансы и статистика, 2000.
29. Хансен Г., Хансен Дж. Базы данных. Разработка и управление : пер. с англ.-Бином,1999.
30. Хеслоп Б.,Боденк А. HTML ссамого начало: Пер. с англ.-СПб.: Питер, 1997.

31. Храмцов П.Б. Лабиринт Internet: Практическое руководства.- М.: ЭЛЕКТРОИНФОРМ ,1996.
32. Экономическая информатика: Учебник/ Под ред. В.В.Евдокимова.-СРб.: Питер,1997.
33. Яккубайтис Э.А. Информационные сети и системы: Справочная книга. - М.: Финансы и статистика, 1996.