ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СТАВРОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

И.П. КУЗЬМЕНКО

ОСНОВЫ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Курс лекций и практических занятий для студентов аграрных вузов, обучающихся по направлению подготовки бакалавров
21.03.02 «Землеустройство и кадастры»,
35.03.07 «Технология производства и переработки сельскохозяйственной продукции»,
19.03.04 «Технология продукции и организация общественного питания»

Ставрополь 2022

Курс лекций и практических занятий для студентов аграрных вузов очной и заочной формы, обучающихся по направлению подготовки бакалавров 21.03.02 «Землеустройство и кадастры», 35.03.07 «Технология производства и переработки сельскохозяйственной продукции», 19.03.04 «Технология продукции и организация общественного питания».

Данный курс обеспечивает студентов знаниями в области современных информационных технологий с учетом требований к уровню подготовки согласно образовательному стандарту. Поможет овладеть основами информационной культуры, современными техническими средства, системным и прикладным программным обеспечением и инструментарием для создания программных продуктов. Освоить технологии работы с основными прикладными пакетами, овладеть принципами работы компьютерных сетей, сетевых средств поиска и обмена информацией.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В настоящее время невозможно быть образованным человеком, не владея основами информационных технологий. Сейчас деятельность людей во многом зависит от их информированности и способностей использовать и обрабатывать информацию. Для свободной ориентации в информационных потоках современный специалист любого профиля должен уметь получать, обрабатывать и использовать информацию с помощью компьютеров, телекоммуникаций и других средств связи.

Очевидно, что основным предметом труда в промышленно развитых странах становится информация. Возникли тенденции неуклонного перекачивания трудовых ресурсов из сферы материального производства в информационную сферу.

В настоящее время уже более половины всего занятого населения наиболее индустриально развитых стран принимают участие в процессе производства и распространения информации.

Информация становится одним из ценнейших ресурсов общества наряду с такими традиционными материальными видами ресурсов, как нефть, газ, полезные ископаемые и др. Появилась новая экономическая категория – национальные информационные ресурсы.

Материальной основой происходящей НТР является электронно-вычислительная техника. На базе этой техники появляется новый вид технологий – информационные. К ним относятся процессы, где «исходным материалом» и «продукцией» (выходом) является информация. Как производственные, так и информационные технологии возникают не спонтанно, а в результате технологизации того или иного социального процесса, т.е. целенаправленного активного воздействия человека на ту или иную область производства и преобразования её на базе машинной техники. Чем шире использование ЭВМ, тем выше их интеллектуальный уровень, тем больше возникает видов информационных технологий, к которым относятся технологии планирования и управления, научных исследований и разработок, экспериментов, проектирования, денежно-кассовых операций, криминалистики, медицины, образования и др.

Информатизация общества приходит на смену компьютеризации общества. Использование современных информационных технологий в сфере управления обеспечивает повышение качества информации, ее точности, объективности, оперативности и, как следствие этого, возможности принятия своевременных управленческих решений.

Целью изучения курса «Основы информационных технологий» является ознакомление с содержанием информационных технологий как составной части информатики, рассмотрение классификации моделей, методов и средств информационных технологий.

ТЕМА 1. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ИХ РАЗВИТИЕ И КЛАССИФИКАЦИЯ

План:

1. Понятие информационной технологии и их классификация.
2. Этапы развития информационных технологий.
3. Инструментарий ИТ. Соотношение между информационными технологиями и системами.
4. Использование информационных технологий.
5. Виды информационных технологий.

Вопрос 1. Понятие информационной технологии и их классификация

Под **технологией** понимается совокупность методов обработки, изготовления, изменения состояния, свойств, формы сырья, материала или полуфабриката, осуществляемых в процессе производства продукции. Когда мы говорим об информационной технологии, в качестве материала выступает информация. В качестве продукта – тоже информация. Но это качественно новая информация о состоянии объекта, процесса или явления. Технология представлена методами и способами работы с информацией персонала и технических устройств.

Информационная технология (ИТ) – это процесс, использующий совокупность средств и методов сбора, обработки и передачи данных для получения нового качества информации по состоянию объекта, процесса или явления.

Цель применения информационных технологий – снижение трудоемкости использования информационных ресурсов. Человечество занималось обработкой информации тысячи лет. Первые информационные технологии основывались на использовании счетов и письменности. Около пятидесяти лет назад началось исключительно быстрое развитие этих технологий, что в первую очередь связано с появлением компьютеров.

В современном обществе понятие – информационная технология – употребляется в связи с использованием компьютеров для обработки информации. Информационные технологии охватывают вычислительную технику, телевидение, радиовещание, технику связи и даже бытовую электронику. Они находят применение в промышленности, торговле, управлении, банковской системе, образовании, здравоохранении, медицине и науке, транспорте и связи, сельском хозяйстве, системе социального обеспечения, служат подспорьем людям различных профессий.

Народы развитых стран осознают, что совершенствование информационных технологий представляет самую важную, хотя дорогостоящую и трудную задачу. В настоящее время создание крупномасштабных информационно-технологических систем является экономически возможным, и это обусловли-

вает появление национальных исследовательских и образовательных программ, призванных стимулировать их разработку.

ИТ в настоящее время можно классифицировать по ряду признаков:

1. По виду технологии обработки информации:
   - текстовые редакторы;
   - электронные таблицы;
   - системы управления базами данных;
   - системы обработки графической информации;
   - системы обработки звуковой информации;
   - мультимедийные системы;
   - экспертные системы и искусственный интеллект;
   - оперативный поиск информации во внешних базах данных;
   - гипертекстовые системы;
   - системы программирования;
   - интегрированные пакеты.
2. По способу реализации ИТ в информационных системах:
   - традиционно сложившиеся;
   - новые информационные технологии.
3. По типу пользовательского интерфейса:
   - с командным интерфейсом;
   - с*WIMP*-интерфейсом;
   - с*SILK*-интерфейсом.
4. По способу построения компьютерной сети:
   - локальные;
   - многоуровневые;
   - распределенные.
5. По области управления социально-экономическим процессом:
   - банковские;
   - налоговые;
   - учетные и аудиторские;
   - финансовые;
   - страховые;
   - управления торговлей;
   - управления производством и хозяйственным процессом;
   - системы регионального управления.

По виду технологии обработки информации ИТ рассматриваются в программном аспекте и включают: текстовую обработку, электронные таблицы, автоматизированные банки данных, обработку графической информации, муль-

тимедийные и другие системы.

Перспективным направлением развития компьютерной технологии является создание программных средств для вывода высококачественного звука и видеоизображения. Технология формирования видеоизображения получила название компьютерной графики. Компьютерная графика – это создание, хранение и обработка моделей объектов и их изображений с помощью компьютера. Данная технология проникла в область экономического анализа, моделирования различного рода конструкций, она незаменима в производстве, проникает в рекламную деятельность, делает занимательным досуг. Формируемые и обрабатываемые с помощью цифрового процессора изображения могут быть демонстрационными и анимационными.

Программно-техническая организация обмена с компьютером текстовой, графической, аудио- и видеоинформацией получила название мультимедиатехнологий. Такую технологию реализуют специальные программные средства, имеющие встроенную поддержку мультимедиа и позволяющие использовать ее в профессиональной деятельности, учебно-образовательных, научно-популярных и игровых областях. При применении этой технологии в экономической работе открываются реальные перспективы использовать компьютер для озвучивания изображений, а также понимания им человеческой речи, ведения компьютером диалога со специалистом на родном для специалиста языке. Способность компьютера с голоса воспринимать несложные команды управления программами, открытием файлов, выводом информации на печать и другими операциями в ближайшем будущем создаст самые благоприятные условия пользователю для взаимодействия с ним в процессе профессиональной деятельности.

При традиционном способе реализации в информационных системах (ИС) ИТ существовали в условиях централизованной обработки данных, до массового использования персональных ЭВМ, и были ориентированы, главным образом, на снижение трудоемкости при формировании регулярной отчетности, то новые информационные технологии связаны с информационным обеспечением процесса управления в режиме реального времени.

Новая информационная технология – технология, которая основывается на применении компьютеров, активном участии пользователей (непрофессионалов в области программирования) в информационном процессе; высоком уровне дружественного пользовательского интерфейса; широком использовании пакетов прикладных программ общего и проблемного назначения; возможности доступа (для пользователя) к удаленным базам данных и программам благодаря вычислительным сетям ЭВМ.

По типу пользовательского интерфейса можно рассматривать ИТ с точки

зрения возможностей доступа пользователя к информационным и вычислительным ресурсам (под интерфейсом понимают определенные стандартом правила взаимодействия пользователей, устройств, программ).

С помощью командного интерфейса пользователь подает команды компьютеру, а компьютер их выполняет и выдает результат пользователю. Командный интерфейс реализован в виде пакетной технологии и технологии командной строки.

Пакетная ИТ исключает возможность пользователя влиять на обработку информации пока она производится в автоматическом режиме. Это объясняется организацией обработки, которая основана на выполнении программно-заданной последовательности операций над заранее накопленными в системе и объединенными в пакет данными.

При использовании технологии командной строки в качестве единственного способа ввода информации от человека к компьютеру служит клавиатура, а компьютер выводит информацию человеку с помощью алфавитно-цифрового дисплея (монитора).

В отличие от пакетной диалоговая ИТ предоставляет пользователю неограниченную возможность взаимодействовать с хранящимися в системе информационными ресурсами в реальном масштабе времени, получая при этом всю необходимую информацию для решения функциональных задач и принятия решений.

Интерфейс сетевой ИТ предоставляет пользователю средства теледоступа к территориально распределенным информационным и вычислительным ресурсам благодаря развитым средствам связи, что делает такие ИТ широко используемыми и многофункциональными.

Характерная особенность *WIMP*-интерфейса (*Window* – окно, *Image* – образ, *Menu* – меню, *Pointer* – указатель) – ведение диалога с пользователем с помощью графических образов – меню, окон, других элементов. Примером программ с графическим интерфейсом является операционная система *MS Windows*.

Существует, но пока не широко используется *SILK*-интерфейс (*Speech* – речь, *Image* – образ, *Language* – язык, *Knowledge* – знание). Он наиболее приближен к обычной, человеческой форме общения. В рамках этого интерфейса идет «разговор» человека и компьютера. Компьютер, анализируя человеческую речь, находит для себя команды, выбирая в ней ключевые фразы. Результат выполнения команд он также преобразует в понятную человеку форму. Разновидностями интерфейсов являются интерфейсы на основе речевой (команды подаются голосом путем произнесения специальных зарезервированных слов – команд) и биометрической технологий (для управления компьютером используется выражение лица человека, направление его взгляда, размер зрачка, рисунок радужной оболочки глаз, отпечатки пальцев и другая уникальная информация). Изображения считываются с цифровой видеокамеры, а затем с помощью

специальных программ распознавания образов из этого изображения выделяются команды.

Повышение требований к оперативности информационного обмена и управления, а следовательно, к срочности обработки информации, привело к созданию не только локальных, но и многоуровневых и распределенных систем организационного управления объектами, какими являются, например, банковские, налоговые, снабженческие, статистические и другие службы. Их информационное обеспечение реализуют сети автоматизированных банков данных, которые строятся с учетом организационно-функциональной структуры соответствующего многоуровневого экономического объекта, машинного ведения информационных массивов. Эту проблему в новых информационных технологиях решают распределенные системы обработки данных с использованием каналов связи для обмена информацией между базами данных различных уровней. За счет усложнения программных средств управления базами данных повышается скорость, обеспечиваются защита и достоверность информации при выполнении экономических расчетов и выработке управленческих решений.

В многоуровневых и распределенных компьютерных информационных системах организационного управления одинаково успешно могут быть решены как проблемы оперативной работы с информацией, так и проблемы анализа экономических ситуаций при выработке и принятии управленческих решений. В частности, создаваемые автоматизированные рабочие места специалистов предоставляют возможность пользователям работать в диалоговом режиме, оперативно решать текущие задачи, удобно вводить данные с терминала, вести их визуальный контроль, вызывать нужную информацию для обработки, определять достоверность результатной информации и выводить ее на экран, печатающее устройство или передавать по каналам связи.

В настоящее время наблюдается тенденция к объединению различных типов информационных технологий в единый компьютерно-технологический комплекс, который носит название интегрированного. Особое место в нем принадлежит средствам телекоммуникации, обеспечивающим не только чрезвычайно широкие технологические возможности автоматизации управленческой деятельности, но и являющимся основой создания самых разнообразных сетевых вариантов ИТ: локальных, многоуровневых, распределенных, глобальных вычислительных сетей, электронной почты, цифровых сетей; интегрального обслуживания. Все они ориентированы на технологическое взаимодействие совокупности объектов, образуемых устройствами передачи, обработки, накопления и хранения, защиты данных; представляют собой интегрированные компьютерные системы обработки данных большой сложности, практически неограниченных эксплуатационных возможностей для реализации управленческих процессов в экономике.

Информационно-телекоммуникационные технологии (ИТТ) в современных организациях играют чрезвычайно важную роль. Они обеспечивают

выполнение самых разных задач:

- доступ к внешним и внутренним базам данных в режиме прямого доступа для получения исследовательской, научной, рабочей и другой информации;
- использование экспертных систем для диагностики, управления и принятия решений;
- передачу данных по электронной почте;
- формирование электронных бюллетеней для деловой и технической информации общего пользования;
- проведение видеоконференций;
- создание систем хранения и поиска информации;
- компьютерный дизайн;
- компьютерное обучение;
- индексацию и хранение документов.

Очень интенсивно на корпоративном уровне используются интранет-технологии, существенно упрощающие работу с большими массивами информации, их структуризацию, поиск и деловое применение. Кредитные организации используют ИТТ для определения финансового риска при инвестициях и операциях с ценными бумагами.

Роль ИТТ в традиционных отраслях промышленности и сфере услуг (транспортные перевозки, туризм, медицинское обслуживание, издательство, страхование, розничная торговля и т.п.) столь велика, что без их использования выдержать острую конкуренцию практически невозможно.

Интегрированные компьютерные системы обработки данных проектируются как сложный информационно-технологический и программный комплекс. Он поддерживает единый способ представления данных и взаимодействие пользователей с компонентами системы, обеспечивает информационные и вычислительные потребности специалистов. Особое значение в таких системах придается защите информации при ее передаче и обработке. Наибольшее распространение при защите экономической информации получили аппаратно-программные способы, например, использование системы связи, выбранной по защитным свойствам и качеству обслуживания, гарантирующим сохранность информации в процессе передачи и доставки ее адресату, шифрование и дешифрование данных абонентами сетей общего пользования (телефонных, телеграфных) при договоренности пользователей об общих технических средствах, алгоритмах шифрования и т.п.

Вопрос 2. Этапы развития информационных технологий

Существует несколько точек зрения относительно развития информацион-

ных технологий с использованием компьютеров, которые определяются различными признаками деления.

Общим для всех изложенных далее подходов является то, что с появлением персонального компьютера начался новый этап развития информационной технологии. Основной целью становится удовлетворение персональных информационных потребностей человека как для профессиональной сферы, так и для бытовой.

Этапы развития информационных технологий, выделяемые по видам задач и процессов обработки информации:

1-й этап (60–70-е гг.) – обработка данных в вычислительных центрах в режиме коллективного пользования. Основным направлением развития информационной технологии являлась автоматизация операционных рутинных действий человека.

2-й этап (с 80-х гг.) – создание информационных технологий, направленных на решение стратегических задач.

Этапы развития информационных технологий, выделяемые по проблемам, стоящим на пути информатизации общества:

1-й этап (до конца 60-х гг.) характеризуется проблемой, связанной с обработкой больших объемов данных в условиях ограниченных возможностей аппаратных средств.

2-й этап (до конца 70-х гг.) связывается с распространением ЭВМ серии *IBM*/360. Проблема этого этапа – отставание программного обеспечения от уровня развития аппаратных средств.

3-й этап (с начала 80-х гг.) – компьютер становится инструментом непрофессионального пользователя, а информационные системы – средством поддержки принятия его решений. Проблемы – максимальное удовлетворение потребностей пользователя и создание соответствующего интерфейса работы в компьютерной среде.

4-й этап (с начала 90-х гг.) – создание современной технологии межорганизационных связей и информационных систем. Проблемы этого этапа весьма многочисленны: выработка соглашений и установление стандартов, протоколов для компьютерной связи; организация доступа к стратегической информации; организация защиты и безопасности информации.

Этапы развития информационных технологий, выделяемые по преимуществам, которые приносит компьютерная технология:

1-й этап (с начала 60-х гг.) характеризуется довольно эффективной обра- боткой информации при выполнении рутинных операций с ориентацией на централизованное коллективное использование ресурсов вычислительных центров. Основным критерием оценки эффективности

создаваемых информационных систем была разница между затраченными на разработку и сэкономленными в результате внедрения средствами. Основной проблемой на этом этапе была психологическая – плохое взаимодействие пользователей, для которых создавались информационные системы, и разработчиков из-за различия их взглядов и понимания решаемых проблем. Как следствие этой проблемы, создавались системы, которые пользователи плохо воспринимали и, несмотря на их достаточно большие возможности, не использовали в полной мере.

1-й этап (с середины 70-х гг.) связан с появлением персональных компьютеров. Изменился подход к созданию информационных систем – ориентация смещается в сторону индивидуального пользователя для поддержки принимаемых им решений. Пользователь заинтересован в проводимой разработке, налаживается контакт с разработчиком, возникает взаимопонимание обеих групп специалистов. На этом этапе используется как централизованная обработка данных, характерная для первого этапа, так и децентрализованная, базирующаяся на решении локальных задач и работе с локальными базами данных на рабочем месте пользователя.

2-й этап (с начала 90-х гг.) связан с понятием анализа стратегических преимуществ в бизнесе и основан на достижениях телекоммуникационной технологии распределенной обработки информации. Информационные системы имеют своей целью не просто увеличение эффективности обработки данных и помощь управленцу. Соответствующие информационные технологии должны помочь организации выстоять в конкурентной борьбе и получить преимущество.

Этапы развития информационных технологий, выделяемые по видам инст рументария технологии:

1-й этап (до второй половины XIX в.) – «ручная» информационная технология, инструментарий которой составляли перо, чернильница, книга. Коммуникации осуществлялись ручным способом путем переправки через почту писем, пакетов, депеш. Основная цель технологии – представление информации в нужной форме.

2-й этап (с конца XIX в.) – «механическая» технология, инструментарий которой составляли: пишущая машинка, телефон, диктофон, оснащенная более совершенными средствами доставки почта. Основная цель технологии – представление информации в нужной форме более удобными средствами.

3-й этап (40 – 60-е гг. XX в.) – «электрическая» технология, инструментарий которой составляли: большие ЭВМ и соответствующее программное обеспечение, электрические пишущие машинки, ксероксы, портативные диктофоны. Изменяется цель технологии: акцент в информационной технологии начинает перемещаться с формы представления информации на формирование ее

содержания.

4-й этап (с начала 70-х гг.) – «электронная» технология, основным инстру- ментарием которой становятся большие ЭВМ и создаваемые на их базе автоматизированные системы управления (АСУ) и информационно-поисковые системы (ИПС), оснащенные широким спектром базовых и специализированных программных комплексов. Центр тяжести технологии еще более смещается на формирование содержательной стороны информации для управленческой среды различных сфер общественной жизни, особенно на организацию аналитической работы. Множество объективных и субъективных факторов не позволило решить стоящие перед новой концепцией информационной технологии поставленные задачи. Однако был приобретен опыт формирования содержательной стороны управленческой информации и подготовлена профессиональная, психологическая и социальная база для перехода на новый этап развития технологии.

4-й этап (с середины 80-х гг.) – «компьютерная» («новая») технология, основным инструментарием которой является персональный компьютер с широким спектром стандартных программных продуктов разного назначения. На этом этапе происходит процесс персонализации АСУ, который проявляется в создании систем поддержки принятия решений определенными специалистами. Подобные системы имеют встроенные элементы анализа и интеллекта для разных уровней управления, реализуются на персональном компьютере и используют телекоммуникации. В связи с переходом на микропроцессорную базу существенным изменениям подвергаются и технические средства бытового, культурного и прочего назначений. Начинают широко использоваться глобальные и локальные компьютерные сети.

Вопрос 3. Инструментарий ИТ. Соотношение между информационными технологиями и системами

Реализация технологического процесса материального производства осуществляется с помощью различных технических средств, к которым относятся: оборудование, станки, инструменты, конвейерные линии и т.п.

По аналогии и для ИТ можно определить что-то подобное. Такими техническими средствами производства информации является аппаратное, программное и математическое обеспечение этого процесса. С их помощью производится переработка первичной информации в информацию нового качества.

Инструментарий ИТ – это один или несколько взаимосвязанных программных продуктов для определенного типа компьютерной техники и техно-

логии работы, позволяющих достичь для каждого пользователя поставленной цели.

В качестве инструментария можно использовать распространенные виды программных продуктов для персонального компьютера: текстовый процессор, электронные таблицы, настольные издательские системы, системы управления базами данных, электронные записные книжки, электронные календари, информационные системы функционального назначения (финансовые, бухгалтерские, налоговые и т.д.) экспертные системы и т.д.

ИТ тесно связано с ИС, которые являются для нее основной средой. ИТ является процессом, состоящим из четко регламентированных правил выполнения операций, действий, этапов разной степени сложности над данными, хранящимися в компьютерах. Основная цель ИТ – в результате целенаправленных действий по переработке первичной информации получить необходимую для пользователя информацию.

ИС является средой, составляющими элементами которой являются компьютеры, компьютерные сети, программные продукты, базы данных, люди, различного рода технические и программные средства связи и т.д. Основная цель ИС – организация хранения и передачи информации. ИС представляет собой человеко-компьютерную систему обработки информации.

Реализация функций ИС невозможна без знания ориентированной на нее ИТ. ИТ может существовать и вне ИС.

Пример ИТ работы в среде текстового процессора *MSWord*, который не является ИС. ИТ мультимедиа, где с помощью телекоммуникационной связи осуществляются передача и обработка на компьютере изображения и звука. Таким образом, ИТ является более емким понятием, отражающим современное представление о процессах преобразования информации в информационном обществе. В умелом сочетании двух ИТ – управленческой и компьютерной – залог успешной работы ИС.

Таким образом, определения ИС и технологий, реализованных средствами компьютерной техники, будут следующими:

- ИТ – совокупность четко определенных целенаправленных действий персонала по переработке информации на компьютере;
- ИС – человеко-компьютерная система для поддержки принятия решений и производства информационных продуктов, использующая компьютерную ИТ.

Вопрос 4. Использование информационных технологий

Централизованная обработка информации на ЭВМ вычислительных центров была первой исторически сложившейся технологией. Создавались крупные вычислительные центры коллективного пользования, оснащенные боль-

шими ЭВМ. Применение таких ЭВМ позволяло обрабатывать большие массивы входной информации и получать на этой основе различные виды информационной продукции, которая затем передавалась пользователям. Такой технологический процесс был обусловлен недостаточным оснащением вычислительной техникой предприятий и организаций.

Достоинства методологии централизованной технологии:

- возможность обращения пользователя к большим массивам информации в виде баз данных и к информационной продукции широкой номенклатуры;
- сравнительная легкость внедрения методологических решений по развитию и совершенствованию информационной технологии благодаря централизованному их принятию.

Недостатки такой методологии очевидны:

- ограниченная ответственность низшего персонала, который не способствует оперативному получению информации пользователем, тем самым препятствуя правильности выработки управленческих решений;
- ограничение возможностей пользователя в процессе получения и использования информации.

Децентрализованная обработка информации связана с появлением в 80-х гг. прошлого столетия персональных компьютеров и развитием средств телекоммуникаций. Она дала пользователю широкие возможности в работе с информацией, не ограничивая его инициатив. Достоинствами такой методологии являются:

- гибкость структуры, обеспечивающая простор инициативам пользователя;
- усиление ответственности низшего звена сотрудников;
- уменьшение потребности в пользовании центральным компьютером и соответственно контроле со стороны вычислительного центра;
- более полная реализация творческого потенциала пользователя благодаря использованию средств компьютерной связи.

Однако эта методология имеет свои недостатки:

- сложность стандартизации из-за большого числа уникальных разработок;
- психологическое неприятие пользователями рекомендуемых вычислительным центром стандартов и готовых программных продуктов;
- неравномерность развития уровня информационной технологии на локальных местах, что в первую очередь определяется уровнем квалифика- ции конкретного работника.

Описанные достоинства и недостатки централизованной и децентрализованной информационной технологии привели к необходимости придерживаться линии разумного применения и того, и другого подхода. Такой подход назвали рациональной методологией. Он имеет ряд отличительных особенностей:

- вычислительный центр должен отвечать за выработку общей стратегии использования информационной технологии, помогать пользователям как в работе, так и в обучении, устанавливать стандарты и определять политику применения программных и технических средств;
- персонал, использующий информационную технологию, должен придерживаться указаний вычислительного центра, осуществлять разработку своих локальных систем и технологий в соответствии с общим планом организации.

Рациональная методология использования информационной технологии позволяет достичь большей гибкости, поддерживать общие стандарты, осуществлять совместимость информационных локальных продуктов, снизить дублирование деятельности и др.

При внедрении информационной технологии на предприятии необходимо выбрать одну из двух основных концепций, отражающих сложившиеся точки зрения на существующую структуру организации и роль в ней компьютерной обработки информации.

Первая концепция ориентируется на существующую структуру предприятия. Информационная технология приспосабливается к организационной структуре, и происходит лишь модернизация методов работы. Коммуникации развиты слабо, рационализируются только рабочие места. Происходит распределение функций между техническими работниками и специалистами. Степень риска от внедрения новой информационной технологии минимальна, так как затраты незначительны и организационная структура предприятияне меняется.

Основной недостаток такой стратегии – необходимость непрерывных изменений формы представления информации, приспособленной к конкретным технологическим методам и техническим средствам.К достоинствам стратегии можно отнести минимальные степень риска и затраты.

Вторая концепция ориентируется на будущую структуру предприятия. Существующая структура будет модернизироваться. Данная стратегия предполагает максимальное развитие коммуникаций и разработку новых организационных взаимосвязей. Продуктивность организационной структуры предприятия возрастает, так как рационально распределяются архивы данных, снижается объем циркулирующей по системным каналам информации и достигается сбалансированность между решаемыми задачами.

К основным ее недостаткам следует отнести существенные затраты на первом этапе, связанном с разработкой общей концепции и обследованием всех подразделений предприятия; наличие психологической напряженности, вызванной предполагаемыми изменениями структуры предприятия и, как следствие, изменениями штатного расписания и должностных обязанностей.

Достоинствами данной стратегии являются рационализация организацион-

ной структуры предприятия; максимальная занятость всех работников; высокийпрофессиональный уровень; интеграция профессиональных функций за счет использования компьютерных сетей.

Новая информационная технология на предприятии должна быть такой, чтобы уровни информации и подсистемы, ее обрабатывающие, связывались между собой единым массивом информации. При этом предъявляются два требования. Во-первых, структура системы переработки информации должна соответствовать распределению полномочий в предприятия. Во-вторых, информация внутри системы должна функционировать так, чтобы достаточно полно отражать уровни управления.

Вопрос 5. Виды информационных технологий

Информационная технология обработки данных предназначена для решения хорошо структурированных задач, по которым имеются необходимые входные данные и известны алгоритмы и другие стандартные процедуры их обработки. Эта технология применяется на уровне операционной (исполнительской) деятельности персонала невысокой квалификации в целях автоматизации некоторых рутинных постоянно повторяющихся операций управленческого труда. Поэтому внедрение информационных технологий и систем на этом уровне существенно повысит производительность труда персонала, освободит его от рутинных операций, возможно, даже приведет к необходимости сокращения численности работников.

На уровне операционной деятельности решаются следующие задачи:

- обработка данных об операциях, производимых на предприятии;
- создание периодических контрольных отчетов о состоянии дел;
- получение ответов на всевозможные текущие запросы и оформление их в виде бумажных документов или отчетов.

Существует несколько особенностей, связанных с обработкой данных, отличающих данную технологию от всех прочих:

- решение только хорошо структурированных задач, для которых можно разработать алгоритм;
- выполнение необходимых задач по обработке данных (обязательно должна быть информационная система обработки данных и разработана соответствующая информационная технология);
- выполнение стандартных процедур обработки;
- выполнение основного объема работ в автоматическом режиме с минимальным участием человека;
- использование детализированных данных;

- акцент на хронологию событий;
- требование минимальной помощи в решении проблем со стороны специалистов других уровней.

Основные компоненты информационной технологии обработки данных:

1) сбор данных – каждое действие на предприятии в результате его функционирования сопровождается соответствующими записями данных;

2) обработка данных – при этом используются следующие типовые операции:

- классификация или группировка, для чего используется метод кодирования записей;
- сортировка, с помощью которой упорядочивается последовательность записей;
- вычисления, включающие арифметические и логические операции, позволяющие получать новые данные;
- укрупнение или агрегирование, служащее для уменьшения количества данных и реализуемое в форме расчетов итоговых или средних значений;

3) хранение данных с использованием баз данных;

4) создание отчетов (документов) – как по запросу или в связи с проведенной операцией, так и периодически в конце каждого месяца, квартала или года.

Целью **информационной технологии управления** является удовлетворение информационных потребностей всех без исключения сотрудников предприятия, имеющих дело с принятием решений. Эта технология ориентирована на работу в среде информационной системы управления и используется при худшей структурированности решаемых задач, если их сравнивать с задачами, решаемыми с помощью информационной технологии обработки данных.

Для принятия решений на уровне управленческого контроля информация должна быть представлена в агрегированном виде так, чтобы просматривались тенденции изменения данных, причины возникших отклонений и возможные решения. На этом этапе решаются следующие задачи обработки данных:

- оценка планируемого состояния объекта управления;
- оценка отклонений от планируемого состояния;
- выявление причин отклонений;
- анализ возможных решений и действий.

Информационная технология управления направлена на создание различных видов отчетов. Регулярные отчеты создаются в соответствии с установленным графиком, определяющим время их создания, например месячный анализ продаж компании. Специальные отчеты создаются по запросам управленцев или когда в компании произошло что-то незапланированное. И те, и другие виды отчетов могут иметь форму суммирующих, сравнительных и чрезвычайных отчетов.Использование отчетов для поддержки управления оказывается

особенно эффективным при реализации так называемого управления по отклонениям, т.е. состояния хозяйственной деятельности предприятияотличается от некоторых установленных стандартов.

Основными компонентами информационной технологии управления будут: входная информация поступает из систем операционного уровня; выходная информация формируется в виде управленческих отчетов в удобном для принятия решения виде; содержимое базы данных при помощи соответствующего программного обеспечения преобразуется в периодические и специальные отчеты, поступающие к специалистам, участвующим в принятии решений в организации.

Автоматизация офиса изначально использовалась на производстве и затем распространилась на офис, имея вначале целью лишь автоматизацию рутинной секретарской работы. По мере развития средств коммуникаций автоматизация офисных технологий заинтересовала специалистов и управленцев, которые увидели в ней возможность повысить производительность своего труда.

Автоматизированный офис привлекателен для менеджеров всех уровней управления на предприятии не только потому, что поддерживает внутреннюю связь персонала, но также потому, что предоставляет им новые средства коммуникации с внешним окружением.

Информационная технология автоматизированного офиса – организация и поддержка коммуникационных процессов как внутри организации, так и с внешней средой на базе компьютерных сетей и других современных средств передачи и работы с информацией.

Офисные автоматизированные технологии используются управленцами, специалистами, секретарями и конторскими служащими, особенно они привлекательны для группового решения проблем. Улучшение принимаемых менеджерами решений в результате их более совершенной коммуникации способно обеспечить экономический рост предприятия.

В настоящее время известно несколько десятков программных продуктов для компьютеров и некомпьютерных технических средств, обеспечивающих технологию автоматизации офиса: текстовый процессор, табличный процессор, электронная почта, электронный календарь, аудиопочта, компьютерные и телеконференции, видеотекст, хранение изображений, а также специализированные программы управленческой деятельности: ведения документов, контроля за исполнением приказов и т.д.

Также широко используются некомпьютерные средства: аудио- и видеоконференции, факсимильная связь, ксерокс и другие средства оргтехники.

Обязательным компонентом любой технологии является база данных. В автоматизированном офисе база данных концентрирует в себе данные о производственной системе предприятия так же, как в технологии обработки данных

на операционном уровне.

Информация в базу данных может также поступать из внешнего окружения предприятия, а из нее на вход компьютерных приложений (программ), таких, как текстовый процессор, табличный процессор, электронная почта, компьютерные конференции и пр. Любое компьютерное приложение автоматизированного офиса обеспечивает работникам связь друг с другом и с другими предприятиями.

Текстовый процессор предназначен для создания и обработки текстовых документов. Когда документ готов, работник переписывает его во внешнюю память, а затем распечатывает и при необходимости передает по компьютерной сети. Таким образом, в распоряжении менеджера имеется эффективный вид письменной коммуникации.

Электронная почта (*E-mail*), основываясь на сетевом использовании компьютеров, дает возможность пользователю получать, хранить и отправлять сообщения своим партнерам по сети. Здесь имеет место только однонаправленная связь. Электронная почта может предоставлять пользователю различные возможности в зависимости от используемого программного обеспечения. Чтобы посылаемое сообщение стало доступно всем пользователям электронной почты, его следует поместить на компьютерную доску объявлений, при желании можно указать, что это частная корреспонденция. Вы также можете послать отправление с уведомлением о его получении адресатом.

Аудиопочта – это почта для передачи сообщений голосом. Она напоминает электронную почту, за исключением того, что вместо набора сообщения на клавиатуре компьютера его передача происходит через телефон. Система включает в себя специальное устройство для преобразования аудиосигналов в цифровой код и обратно, а также компьютер для хранения аудиосообщений в цифровой форме. Главным преимуществом аудиопочты по сравнению с электронной является то, что при ее использовании не нужно вводить данные с клавиатуры.

Табличный процессор, так же, как и текстовый, является базовой составляющей информационной культуры любого сотрудника и автоматизированной офисной технологии. Без знания основ технологии работы в нем невозможно полноценно использовать персональный компьютер в своей деятельности. Функции современных программных сред табличных процессоров позволяют выполнять многочисленные операции над данными, представленными в табличной форме. Объединяя эти операции по общим признакам, можно выделить наиболее многочисленные и применяемые группы технологических операций:

- ввод данных как с клавиатуры, так и из баз данных;
- обработка данных (сортировка, автоматическое формирование итогов,копирование и перенос данных, различные группы операций по

вычислениям, агрегирование данных и т.д.);

- вывод информации в печатном виде, в виде импортируемых файлов в другие системы непосредственно в базу данных;
- качественное оформление табличных форм представления данных;
- многоплановое и качественное оформление данных в виде диаграмм и графиков;
- проведение инженерных, финансовых, статистических расчетов;
- проведение математического моделирования и ряд других вспомогательных операций.

Электронный календарь предоставляет еще одну возможность использовать сетевой вариант компьютера для хранения и манипулирования рабочим расписанием управленцев и других работников организации. Менеджер (или его секретарь) устанавливает дату и время встречи или другого мероприятия, просматривает получившееся расписание, вносит изменения при помощи клавиатуры. Техническое и программное обеспечение электронного календаря полностью соответствует аналогичным компонентам электронной почты. Более того, программное обеспечение календаря часто является составной частью программного обеспечения электронной почты.

Компьютерные конференции используют компьютерные сети для обмена информацией между участниками группы, решающей определенную проблему. Естественно, круг лиц, имеющих доступ к этой технологии, ограничен. Количество участников компьютерной конференции может быть во много раз больше, чем аудио- и видеоконференций. В литературе часто можно встретить термин телеконференция. Телеконференция включает в себя три типа конференций: аудио, видео и компьютерную.

Видеотекст основан на использовании компьютера для получения отображения текстовых и графических данных на экране монитора. Для лиц, принимающих решение, имеются три возможности получения информации в форме видеотекста:

- создать файлы видеотекста на своих собственных компьютерах;
- заключить договор со специализированной компанией на получение доступа к разработанным ею файлам видеотекста. Такие файлы, специально предназначенные для продажи, могут храниться на серверах компании, осуществляющей подобные услуги, или поставляться клиенту на магнитных или оптических дисках;
- заключить договоры с другими компаниями на получение доступа к их файлам видео текста.

Обмен каталогами и ценниками (прайс-листами) своей продукции между компаниями в форме видеотекста приобретает сейчас все большую популяр-

ность. Что же касается компаний, специализирующихся на продаже видеотекста, то их услуги начинают конкурировать с такой печатной продукцией, как газеты и журналы.

На любом предприятии необходимо длительное время хранить большое количество документов. Их число может быть так велико, что хранение даже в форме файлов вызывает серьезные проблемы. Поэтому возникла идея хранить не сам документ, а его образ (изображение), причем хранить в цифровой форме. Хранение изображений (*imaging*) является перспективной офисной технологией и основывается на использовании специального устройства – оптического распознавателя образов, позволяющего преобразовывать изображение документа или фильма в цифровой вид для дальнейшего хранения во внешней памяти компьютера. Сохраненное в цифровом формате изображение может быть в любой момент выведено в его реальном виде на экран или принтер. Для хранения изображений используются оптические диски, обладающие огромными емкостями. Так, на пятидюймовый оптический диск можно записать около 200 тыс. страниц.

Аудиоконференции используют аудиосвязь для поддержания коммуникаций между территориально удаленными работниками или подразделениями предприятия. Наиболее простым техническим средством реализации аудиоконференций является телефонная связь, оснащенная дополнительными устройствами, дающими возможность участия в разговоре более чем двум участникам. Создание аудиоконференций не требует наличия компьютера, а лишь предполагает использование двухсторонней аудиосвязи между ее участниками. Использование аудиоконференций облегчает принятие решений, оно дешево и удобно.

Видеоконференции предназначены для тех же целей, что и аудиоконференции, с применением видеоаппаратуры. Их проведение не требует компьютера. В процессе видеоконференции ее участники, удаленные друг от друга на значительное расстояние, могут видеть на телевизионном экране себя и других участников. Одновременно с телевизионным изображением передается звуковое сопровождение. Хотя видеоконференции позволяют сократить транспортные и командировочные расходы, большинство предприятий применяет их не только по этой причине. Эти предприятия видят в них возможность привлечь к решению проблем максимальное количество менеджеров и других работников, территориально удаленных от главного офиса. Наиболее популярны три конфигурации построения видеоконференций:

- односторонняя видео- и аудиосвязь (видео- и аудиосигналы идут только в одном направлении, например от руководителя проекта к исполнителям);
- односторонняя видео- и двухсторонняя аудиосвязь (двухсторонняя аудиосвязь дает возможность участникам конференции, принимающим видеоизображение, обмениваться аудиоинформацией с передающим

видеосигнал участником);

- двухсторонняя видео- и аудиосвязь.

Факсимильная связь основана на использовании факс-аппарата, способного читать документ на одном конце коммуникационного канала и воспроизводить его изображение на другом. Факсимильная связь вносит свой вклад в принятие решений за счет быстрой и легкой рассылки документов участникам группы, решающей определенную проблему, независимо от их географического положения.

Главной особенностью **информационной технологии поддержки принятия решений** является качественно новый метод организации взаимодействия человека и компьютера. Выработка решения, что является основной целью этой технологии, происходит в результате итерационного процесса, в котором участвуют:

- система поддержки принятия решений в роли вычислительного звена и объекта управления;
- человек как управляющее звено, задающее входные данные и оценивающее полученный результат вычислений на компьютере.

Кроме того, что информационная технология поддержки принятия решений способна совместно с пользователем создавать новую информацию для принятия решений, можно указать еще ряд ее отличительных характеристик:

- ориентация на решение плохо структурированных (формализованных) задач;
- сочетание традиционных методов доступа и обработки компьютерных данных с возможностями математических моделей и методами решения задач на их основе;
- направленность на непрофессионального пользователя компьютера;
- высокая адаптивность, обеспечивающая возможность приспосабливаться к особенностям имеющегося технического и программного обеспечения, а также требованиям пользователя.

В состав системы поддержки принятия решений входят три главных компонента: база данных, база моделей и программная подсистема, которая состоит из системы управления базой данных (СУБД), системы управления базой моделей (СУБМ) и системы управления интерфейсом между пользователем и компьютером.

Например, система поддержки принятия решений, предназначенная для обслуживания клиентов в банке, с помощью мультипликационных моделей может реально просмотреть различные варианты организации обслуживания в зависимости от потока посетителей, допустимой длины очереди, количества пунктов обслуживания и т.п.

Наибольший прогресс среди компьютерных информационных систем отмечен в области разработки **экспертных систем**, основанных на использовании искусственного интеллекта. Экспертные системы дают возможность менеджеру или специалисту получать консультации экспертов по любым проблемам, о которых этими системами накоплены знания.

Под искусственным интеллектом обычно понимают способности компьютерных систем к таким действиям, которые назывались бы интеллектуальными, если бы исходили от человека. Чаще всего здесь имеются в виду способности, связанные с человеческим мышлением. Работы в области искусственного интеллекта не ограничиваются экспертными системами. Они также включают в себя создание роботов, систем, моделирующих нервную систему человека, его слух, зрение, обоняние, способность к обучению.

Главная идея использования технологии экспертных систем заключается в том, чтобы получить от эксперта его знания и, загрузив их в память компьютера, использовать всякий раз, когда в этом возникнет необходимость. Являясь одним из основных приложений искусственного интеллекта, экспертные системы представляют собой компьютерные программы, трансформирующие опыт экспертов в какой-либо области знаний в форму эвристических правил (эвристик). Эвристики не гарантируют получения оптимального результата с такой же уверенностью, как обычные алгоритмы, используемые для решения задач в рамках технологии поддержки принятия решений. Однако часто они дают в достаточной степени приемлемые решения для их практического использования. Все это делает возможным использовать технологию экспертных систем в качестве советующих систем.

Основными компонентами информационной технологии, используемой в экспертной системе, являются интерфейс пользователя, база знаний, интерпретатор, модуль создания системы.

Менеджер (специалист) использует интерфейс для ввода информации и команд в экспертную систему и получения выходной информации из нее. Технология экспертных систем предусматривает возможность получать в качестве выходной информации не только решение, но и необходимые объяснения. Хотя технология работы с экспертной системой не является простой, пользовательский интерфейс этих систем является дружественным и обычно не вызывает трудностей при ведении диалога.

База знаний содержит факты, описывающие проблемную область, а также логическую взаимосвязь этих фактов. Центральное место в базе знаний принадлежит правилам. Правило определяет, что следует делать в данной конкретной ситуации, и состоит из двух частей: условия, которое может выполняться или нет, и действия, которое следует произвести, если условие выполняется.

Интерпретатор – часть экспертной системы, производящая в определенном порядке обработку знаний (мышление), находящихся в базе знаний. Технология работы интерпретатора сводится к последовательному рассмотрению совокупности правил (правило за правилом). Если условие, содержащееся в правиле, соблюдается, выполняется определенное действие, и пользователю предоставляется вариант решения его проблемы. Кроме того, во многих экспертных системах вводятся дополнительные блоки: база данных, блок расчета, блок ввода и корректировки данных.

Модуль создания системы служит для создания набора (иерархии) правил. Существуют два подхода, которые могут быть положены в основу модуля создания системы: использование алгоритмических языков программирования и использование оболочек экспертных систем.

Оболочка экспертных систем представляет собой готовую программную среду, которая может быть приспособлена к решению определенной проблемы путем создания соответствующей базы знаний. В большинстве случаев использование оболочек позволяет создавать экспертные системы быстрее и легче в сравнении с программированием.

ТЕМА 2. ТЕХНИЧЕСКИЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

План:

1. Общие понятия вычислительной системы.
2. Аппаратная конфигурация вычислительных систем.

Вопрос 1. Общие понятия вычислительной системы

Состав вычислительной системы называется конфигурацией. В состав вычислительной системы входит:

- *hardware* – аппаратное обеспечение компьютера;
- *software* – программное обеспечение компьютера;
- *brainware* – интеллектуальное обеспечение компьютера, то есть человек, работающий с компьютером.

Качественное изучение первых двух составляющих приводит к тому, что обучаемый может стать полноценной третьей составляющей вычислительной системы.

Архитектура ЭВМ – это множество ресурсов, доступных пользователю. Архитектура включает в себя: разрядность слова, форматы и систему команд, режимы адресации ОП, состав программно-доступных регистров, объем ОЗУ, способ адресации внешних устройств, слово-состояние процессора и т.д. К наиболее существенным свойствам архитектуры и характеристикам ЭВМ общего назначения можно отнести:

1. Универсальность обеспечивает возможность одинаково решать задачи различных классов практически для всех областей деятельности. Это достигается, прежде всего:

- универсальной системой команд, содержащей кроме операций двоичной арифметики полный набор операций десятичной арифметики с операндами (т.е. элементами данных, над которыми выполняется операция);
- универсальной логической структурой, имеющей обязательные (стандартные) аппаратные и программные средства для всех моделей ЭВМ, образующих единое семейство;
- сбалансированностью входящих в нее устройств по быстродействию и потокам информации между ними.

2. Совместимость достигается аппаратно-программными средствами с целью создания единого прикладного и системного программного обеспечения для всех моделей ЭВМ общего назначения одного семейства. За счет совмести-

мости обеспечивается одинаковость результатов программ и перенос программных средств между различными моделями ЭВМ. Достижение полной совместимости (абсолютной) представляется очень сложной задачей, поэтому в большинстве случаев ограничиваются частичной совместимостью.

3. Развитие программного обеспечения, ориентированного на конкретные структурные и функциональные возможности аппаратуры, позволяющие эффективно решать задачи пользователя.

4. Агрегатный принцип построения технических средств, стандартный интерфейс ввода-вывода, позволяющий подключать различные по назначению периферийные устройства (ПУ) широкой номенклатуры; в совокупности с программным обеспечением позволяют строить конкретный вычислительный комплекс, наиболее подходящий для заданного применения с учетом требований и производительности, функциональным возможностям и набору ПУ.

5. Высокая технологичность обеспечивает возможность крупносерийного производства и высокую технико-экономическую эффективность ЭВМ общего назначения.

6. Соответствие стандартам позволяет обеспечить совместимость с мировым парком ЭВМ общего назначения в части представления информации, способов сопряжения и организации обмена данными.

Аппаратные и программные средства вычислительных систем принято рассматривать отдельно. Такой принцип разделения имеет для информатики особое значение, поскольку очень часто решение одних и тех же задач может обеспечиваться как аппаратными, так и программными средствами.

Критериями выбора аппаратного или программного решения являются производительность и эффективность.

Вопрос 2. Аппаратная конфигурация вычислительных систем

К аппаратному обеспечению вычислительных систем относятся устройства и приборы, образующие аппаратную конфигурацию. Современные компьютеры имеют блочно-модульную конструкцию (совокупность готовых узлов и блоков).

По способу расположения устройств относительно центрального процессорного устройства различают внутренние и внешние (периферийные) устройства.

Согласование между отдельными узлами и блоками выполняют с помощью переходных аппаратно-логических устройств (аппаратных интерфейсов). Стандарты на аппаратные интерфейсы в вычислительной технике называют протоколами. Протокол – это совокупность технических условий, которые должны быть обеспечены разработчиками устройств для успешного согласования их работы с другими устройствами. Многочисленные интерфейсы, присутствующие в архитектуре любой вычислительной системе, можно разделить на

параллельные (передача данных группой битов) и последовательные (передача данных бит за битом). Количество битов в передаваемой группе определяется разрядностью интерфейса (восьмиразрядный интерфейс передает 1 байт (8 бит) за один цикл). Производительность параллельных интерфейсов измеряют байтами в секунду (байт/с, Кбайт/с и т.д.), а последовательных – битами в секунду (бит/с, Кбит/с и т.д.)

Персональные ЭВМ строятся на основе модульной конструкции, которая включает набор конструктивно законченных модулей:

- системный модуль – конструктивно размещенные на одной плате центральный процессор, основная память и разъемы для подключения функциональных модулей;
- функциональные модули – конструктивно размещенные на одной плате контроллеры, адаптеры и дополнительная память, подключаемые к разъемам системного модуля.

Системный и функциональный модули совместно с блоком питания и некоторыми внешними устройствами конструктивно объединяются в единый системный блок, к которому через соответствующие разъемы подключаются выносные внешне устройства: печатающие и клавишное устройства, дисплеи и т.д.

Типовой состав устройств персональной ЭВМвключает системный блок обработки и управления, средства взаимодействия пользователей с системным блоком (монитор, клавиатура, мышь, принтер и т.д.), средства долговременного хранения и накопления данных и средства подключения к каналам связи (рис.1). Такой состав устройств предоставляет в распоряжение индивидуальных пользователей самые разнообразные функциональные возможности.

Рис. 1. Базовая конфигурация вычислительной системы

Системный блок представляет собой основной узел, внутри которого установлены наиболее важные компоненты (рис.2). Устройства, находящиеся внутри него, называются внутренними, а подключаемые снаружи – внешними или периферийными. По внешнему виду системные блоки различаются формой корпуса: *desktop* – в горизонтальном и *tower* – в вертикальном исполнении.

Системный блок – основная часть ПК, в которой установлены устройства,

образующие основную вычислительную мощь компьютера. Они выполнены в виде печатных плат, которые через специальные разъемы (слоты) устанавливаются на одну самую большую – главную плату. Эта плата называется материнской или системной. Также системный блок через различные интерфейсы ввода/вывода объединяет все прочие элементы ПК.

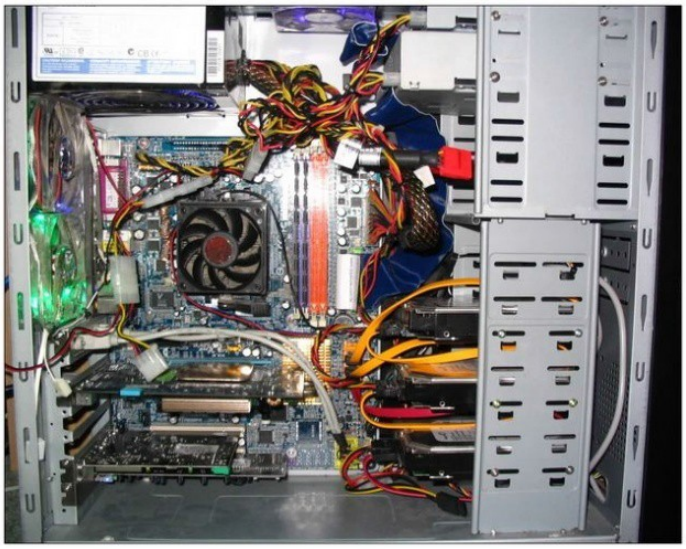


Рис. 2. Внутренняя структура системного блока

Устройства, которые обязательно должны присутствовать в системном блоке компьютера:

Материнская плата – это сложная многослойная печатная плата, на которой устанавливаются основные компоненты персонального компьютера. Как правило, материнская плата содержит разъемы для подключения дополнительных контроллеров, для подключения которых обычно используются шины *USB*, *PCI*,*PCI-Express* , *SATA* и т.п.

Центральный процессор – основная микросхема компьютера, выполняющая программный код, находящийся в памяти и руководит работой устройств компьютера. Работа процессора состоит в выборе из памяти в определенной последовательности команд и данных для их последующего выполнения. Внутренние ячейки процессора называются **регистрами**. Управляя засылкой данных в разные регистры процессора, можно управлять обработкой данных, т.к. разные регистры выполняют разные задачи. Сам процессор также воспринимает специальные команды, которые управляют его действиями. Эти команды ему посылают программы, установленные на компьютере.

В настоящее время производством процессоров занимаются два основных производителя, это компании *Intel* и *AMD* (рис. 3).

Основной характеристикой процессора, показывающей его быстродействие, является так называемая тактовая частота. Она измеряется в гигагерцах (ГГц). Кроме тактовой частоты к параметрам процессоров относятся: рабочее

напряжение, разрядность, коэффициент внутреннего умножения тактовой частоты и размер кэш-памяти.

Рис. 3. Виды центрального процессора

В настоящее время чаще всего производятся многоядерные процессоры, содержащие два, четыре или больше ядер. Многоядерность означает то, что в одном центральном процессоре содержатся как бы два, четыре или больше подпроцессоров, поэтому такие процессоры при той же тактовой частоте (что и одноядерные) дают большую производительность, так как могут выполнять несколько операций параллельно.

Основная память включает оперативное запоминающее устройство (ОЗУ) и постоянное запоминающее устройство (ПЗУ), используемое для хранения программ и данных. ОЗУ персональных ЭВМ или оперативная память являются энергонезависимыми запоминающими устройствами, у которых информация разрушается при отключении питания. Информация, размещаемая в ПЗУ, записывается при изготовлении персональных ЭВМ и не изменяется в течение всего периода ее эксплуатации. В ПЗУ обычно размещаются системные программы, обеспечивающие подготовку персональных ЭВМ к работе после включения питания, т.е. инициализацию (приведение в исходное состояние функциональных модулей), тестирование (проверка работоспособности функциональных модулей) и загрузку оперативной системы.

Оперативная память – это часть системы памяти ПК, предназначена для временного хранения данных и команд, необходимых процессору для выполнения им операций (рис. 4). Оперативная память передает процессору команды и данные непосредственно, либо через кэш-память (буфер). Каждая ячейка оперативной памяти имеет свой индивидуальный адрес.

Рис. 4. Оперативная память

В современных персональных компьютерах оперативная память выполнена по технологии динамической памяти с произвольным доступом. Это понятие предполагает, что текущее обращение к памяти не учитывает порядок предыдущих операций и расположения данных в ней.

Средства долговременного хранения и накопления данных обеспечивают запись и чтение больших массивов информации, в качестве которых могут использоваться: программы в машинных кодах, файлы с данными и т.д.

Отличительные особенности накопителей на жестких дисках типа «винчестер» – герметично закрытая единая конструкция диска.

Жесткий диск – это энергонезависимое, перезаписываемое запоминающее устройство. В настоящий момент в персональных компьютерах применяются два типа жестких дисков:

HDD (англ. harddiskdrive) – запоминающее устройство произвольного доступа, основанное на принципе магнитной записи (рис. 5). Является основным накопителем данных в большинстве компьютеров. В жестких дисках информация хранится на вращающейся металлической или стеклянной пластине, покрытой магнитным материалом. Как правило, диск состоит из нескольких пластин, соединенных общим стержнем. Информация обычно хранится с обеих сторон пластины.

Рис. 5. Жесткий диск типа *HDD*

- *SSD* (англ. *solid-statedrive*) – твердотельный накопитель, энергонезависимое, перезаписываемое запоминающее устройство без движущихся механических частей с использованием флэш-памяти (рис. 6). К преимуществам *SSD* над *HDD* относятся: улучшенные показатели скорости чтения/записи, практически нулевое время для поиска нужной информации, что имеет огромное значение при работе с большим числом небольших по размеру файлов. Также к преимуществам *SSD* относится малый вес и на порядок меньшее энергопотребление, что особенно важно при использовании в мобильных ПК.

К основным параметрам жестких дисков относятся емкость и производительность.

Рис. 6. Жесткий диск типа *SSD*

Без перечисленных устройств персональный компьютер работать не будет. Это обязательные устройства, которые должны быть в любом компьютере, но кроме этих обязательных устройств в компьютер могут быть установлены и другие, дополнительные платы, например, звуковая карта, модем или сетевая карта. Дополнительные устройства только расширяют возможности компьютера, а вот без обязательных устройств компьютер попросту не будет работать.

Видеокарта – это устройство, преобразующее изображение, находящееся в памяти компьютера, в видеосигнал для монитора (рис. 7). Обычно видеокарта является платой расширения и вставляется в специальный разъем для видеокарт на материнской плате (*PCI* или *PCI-Express*), но бывает и встроенной в материнскую плату. Видеокарты имеют встроенный графический процессор, который производит обработку видеоинформации, не нагружая центральный процессор компьютера.

Рис. 7. Видеокарта

К параметрам видеоподсистемы (видеокарта и монитор) относятся разрешение экрана (640×480, 800×600, 1024×768, 1280×1024), цветовое разрешение – количество оттенков, которые может принимать отдельная точка экрана (256, 65 тыс., 16,7 млн цветов), видеоускорение (ускорители 2*D* и 3*D* графики).

Звуковая карта (аудиоадаптер) – это специальная электронная плата, которая позволяет записывать звук, воспроизводить его и создавать программными средствами с помощью микрофона, наушников, динамиков, встроенного синтезатора и другого оборудования.

Основным параметром звуковой карты является разрядность, определяющая количество битов, используемых при преобразовании сигналов из аналого-

вой в цифровую форму и наоборот.

Звуковые карты бывают:

- **Мультимедийные.** Обладают собственным звуковым процессором, который занимается обработкой звука, расчетом трехмерных звуковых эффектов, микшированием звуковых потоков и т.п., что позволяет разгрузить центральный процессор компьютера для обработки более важных задач;

- **Полупрофессиональные.** Отличаются профессиональными схемотехническими решениями и высоким качеством воспроизведения звука, при этом в них, как правило, не используются серьезные звуковые процессоры, поэтому обработку 3D-звука выполняет центральный процессор;

- **Профессиональные.** Характеризуются высоким качеством воспроизведения и записи звука, минимумом искажений, максимумом возможностей для работы с профессиональным программным обеспечением и подключением профессионального оборудования.

В очень многих офисных и бюджетных современных компьютерах в материнскую плату интегрированы следующие устройства: видеокарта; звуковая карта; сетевая карта. Такая интеграция позволяет значительно удешевить ПК при производстве или сделать компьютер с минимальными габаритными размерами, так как отдельные платы занимают определенный объем в корпусе компьютера.

В качестве внешних запоминающих устройств в персональных ЭВМ в основном используются постоянные запоминающие устройства на основе компакт- или *DVD*-дисков, которые вставляются в специальные накопители – дисководы или приводы оптических дисков (рис. 8). **Оптический привод** – электрическое устройство для считывания и записи информации с оптических носителей с помощью сфокусированного лазерного луча. (*CD-ROM*, *DVD-ROM* и т.д.).

Внутренние оптические приводы крепятся внутри системного блока. Обычно бывают полноразмерными (для отсеков 5,25' системного блока) и *slim* (для ноутбуков). Внешние оптическиеприводы располагаются за пределами системного блока, и предназначены в основном для ноутбуков, подключаются к *USB* разъемам.

Основным параметром компакт- или *DVD*-дисков является скорость чтения и записи данных.

Средства взаимодействия пользователей с системным блоком обработки и управления включают устройства ввода-вывода, хранения и обмена данных, обеспечивающие диалоговый обмен информацией (диалоговые внешние устройства).

Рис. 8. Привод оптических дисков

В настоящий момент существует огромный перечень внешних (периферийных) устройств, которые можно подключить к персональному компьютеру. Дело в том, что периферийные устройства расширяют возможности компьютера и поскольку компьютеры широко используются в различных областях деятельности людей, то и периферийные устройства могут быть весьма специфическими. К этому типу внешних периферийных устройств относятся монитор, клавишное устройство (КЛУ), мышь, устройства ввода графических данных, печатающие устройства (ПУ), устройства хранения данных, устройства обмена данными.

Внешние устройства подключаются к компьютеру через специальные **разъемы-порты ввода-вывода**. Наиболее распространены следующие типы портов ввода-вывода:

- *COM* (англ. *communications port*) – последовательный порт; обычно к нему подключаются мышь, модем и другие устройства;

- *LPT* (англ. *Line Print Terminal*) – параллельный порт; обычно используется для подключения принтеров;

- *USB* (англ. *UniversalSerialBus*– «универсальная последовательная шина») – позволяет подключать разнообразные внешние устройства, на- пример, носители информации, клавиатуры/мыши, принтеры, сканеры и многое другое;

- *IEEE* 1394 – последовательная высокоскоростная шина, изначально предназначалась для передачи видеопотока, но благодаря своим прекрасным скоростным характеристикам используется производителями всевозможных устройств, например, внешних жестких дисков или мультимедийных устройств.

Порты ввода/вывода *COM* и *LPT*, устаревшие и с низкой скоростью передачи данных, в настоящее время применяются крайне редко (в основном в промышленности). Со временем шина *USB* нашла наиболее широкое применение, в то время как *IEEE* 1394 была распространена в гораздо меньшей степени. К тому же стандарт *USB* 3.0 вплотную приблизился по скорости передачи данных к *IEEE* 1394. Вследствии этого наибольшее распространение получили устройства с портом ввода/вывода типа *USB*.

Монитор – устройство для визуального отображения информациикомпью-

тером (рис. 9).Это главное устройство вывода. К основным параметрам, на которые следует обратить внимание, относятся диагональ экрана (в дюймах 15″, 17″, 19″ и т.д.), разрешение монитора(1024×768, 1280×1024) и максимальная частота регенерации изображения (частота кадров). Частота регенерации (обновления) изображения показывает, сколько раз в течение секунды монитор может полностью сменить изображение, это также зависит от настроек видеоадаптера.

Рис. 9. Монитор

Имеется два формата мониторов: 5:4 и 16:9 (широкоформатный).

По принципу работы мониторы делят на: ЭЛТ-мониторы (мониторы с электроннолучевой трубкой), ЖК-мониторы (жидкокристаллические), *OLED*-мониторы (органические светодиодные) и *PDP*-мониторы (плазменные).

Изображение на ЭЛТ-мониторе формируется с помощью пучков электронов, которые пускает электроннолучевая трубка. Высокое электрическое напряжение разгоняет эти электроны. Они попадают на поверхность экрана (с внутренней стороны), которая покрыта люминофором. В настоящее время редко применяются из-за ряда недостатков, таких как большие габариты, излучение и тепловыделение в больших количествах, значительная энергоемкость.

Жидкокристаллические мониторы (ЖК-мониторы) также известны как *LCD*-мониторы (*Liquid Crystal Display*). Каждый пиксел ЖК-дисплея состоит из слоя молекул между двумя прозрачными электродами, и двух поляризационных фильтров, плоскости поляризации которых (как правило) перпендикулярны. В отсутствие жидких кристаллов свет, пропускаемый первым фильтром, практически полностью блокируется вторым.

Органические светодиодные мониторы (Organic Leds) – данные мониторы используют органические тонкопленочные материалы, излучающие свет. Данный тип мониторов обеспечивает более широкий спектр цветов и более эффективно использует потребляемую энергию, чем LCD-мониторы. Мониторы такого типа используются в мобильных телефонах, а не в компьютерах.

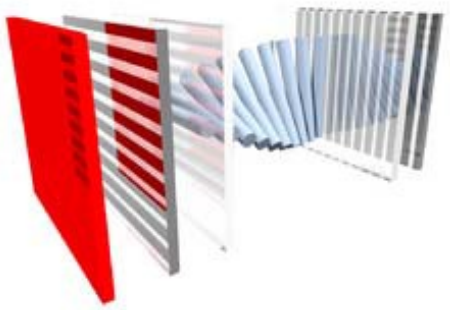


Рис. 10. Принцип работы ЖК-дисплея

Основной недостаток этого вида – малая наработка на отказ по сравнению с другими ви- дами мониторов. Первые образцы в непрерывном режиме могли наработать не более 200 часов, правда, сейчас этот показатель существенно увеличен, но все еще не доходит до показателей тех же *LCD*-мониторов.

Плазменные мониторы (*Plasma Display Panel*) – изображение формируется путем рекомбинации ионизированного газа, в результате чего происходит световой разряд.

Клавишное устройство является основным устройством ввода информации, оно обеспечивает диалоговое общение пользователя с персональной ЭВМ (рис. 11). Клавишное устройство включает в себя клавиатуру и электронный блок кодирования символов клавиатуры. Стандартная клавиатура имеет более 100 клавиш, функционально распределенных по нескольким группам:

1) группа алфавитно-цифровых клавиш предназначена для ввода знаковой информации и команд, набираемых буквами;

2) группа функциональных клавиш (от *F*1 до *F*12);

3) служебные клавиши – ими приходится часто пользоваться (*Shift*, *Alt*, *Ctrl* – в комбинации с другими клавишами, *Enter* – завершение ввода команды, *Tab* – ввод позиции табуляции при наборе текста, *Esc* – отказ от исполнения начатой команды, *Backspace* – удаление введенных знаков, *PrintScreen* – печать текущего состояния экрана или сохранение в буфере обмена, *ScrollLock* – переключение режима работы в некоторых программах, *Pause*/*Break* – приостановка/прерывание текущего процесса);

4) клавиши управления курсором (стрелки, *PageUp*/*PageDown* – перевод курсора на одну страницу вверх или вниз, *Home* и *End* – перевод курсора в начало или конец текущей строки, *Insert* – переключение между режимами вставки и замены, *Delete* – удаление знаков, находящихся справа от текущего положения курсора);

5) группа клавиш дополнительной панели – дублируют действие цифровых и некоторых знаковых клавиш основной панели.

Рис. 11. Клавиатура

Компьютерная мышь – наряду с клавиатурой мышь является важнейшим средством ввода информации. Мышь – устройство управления манипуляторного типа. Перемещение мыши по плоской поверхности синхронизировано с перемещением графического объекта на экране монитора (указателя мыши).

В современных программных продуктах, имеющих сложную графическую оболочку, мышь является основным инструментом управления программой. Одной из важных характеристик мыши является ее разрешение, которое измеряется в *dpi*. Разрешение определяет минимальное перемещение, которое способен почувствовать контроллер мыши. Чем больше разрешение, тем точнее позиционируется мышь, тем с более мелкими объектами можно работать. Нормальное разрешение мыши лежит в диапазоне от 300 до 900 *dpi*. В усовершенствованных мышах используют переменный баллистический эффект скорости, заключающийся в том, что при небольших перемещениях скорость смещения курсора – небольшая, а при значительных перемещениях – существенно увеличивается. Это позволяет эффективнее работать в графических пакетах, где приходится обрабатывать мелкие детали.

Виды мышей по устройствам считывания:

- шариковая (механическая) мышь – принцип работы строится на шарике, который при соприкосновении с поверхностью совокупно запускает 2 ролика, а они уже непосредственно передают информацию по перемещению;
- оптическая мышь (1 поколение) – оптические датчики отслеживают перемещение рабочей области относительно мыши, то есть специальное устройств, которое встроено в мышку, очень часто передает сигнал к коврику и получает обратный, после чего компьютер обрабатывает полученную информацию и выводит на экран в виде готовых действий;
- оптическая мышь (2 поколение) – процесс движениясхож процессом мыши 1 поколения, но отпала необходимость в специальном коврике, компьютер сам определяет координаты и выполняет перемещение курсора;
- лазерная компьютерная мышь использует тот же принцип работы, что и оп- тическая, но для подсветки использует уже лазер, а не светодиод. Они не так

требовательны к поверхности, а также более точно оценивает перемещение;

- гироскопические мыши – мыши, которым не требуется рабочей поверхности, работа с ними ведется через передвижение аппарата в воздухе. Мыши также разделяются на проводные и беспроводные.

К устройствам ввода графических данных относятся сканеры, дигитайзеры и цифровые фото- и видеокамеры.

Сканер – это устройство, которое анализируя какой-либо объект (обычно изображение, текст), создает цифровую копию изображения объекта (рис. 12). Процесс получения этой копии называется сканированием.

Рис. 12. Сканерпланшетный

К основным характеристикам сканеров относят: оптическое разрешение (нормальный уровень разрешения не менее 600 *dpi*); разрешение по *X* и *Y* (определяет общее качество отсканированного изображения); скорость сканирования (количество сканированиястраниц в минуту или время, необходимое для сканирования одной страницы); глубина цвета; максимальная оптическая плотность; тип источника света (ксеноновые, флуоресцентные лампы, светодиоды); тип датчика сканера.

Виды сканеров:

- планшетные – планшет, внутри которого под прозрачным стеклом расположен механизм сканирования;
- ручные – в них отсутствует двигатель, следовательно, объект приходится сканировать пользователю вручную, к плюсам можно отнести дешевизну и мобильность;
- листопротяжные – лист бумаги вставляется в щель и протягивается по направляющим роликам внутри сканера мимо лампы. Имеет меньшие размеры, по сравнению с планшетным, однако может сканировать только отдельные листы;
- планетарные сканеры – применяются для сканирования книг или легко повреждающихся документов, так как при сканировании нет контакта со сканируемым объектом;
- книжные сканеры – предназначены для сканирования брошюрованных документов лицевой стороной вверх;

- слайд-сканеры – служат для сканирования пленочных слайдов, выпускаются как самостоятельные устройства, так и в виде дополнительных модулей к обычным сканерам.
- сканеры штрих-кода – небольшие, компактные модели для сканирования штрих-кодов товара в магазинах.

Дигитайзер (планшет) – одно устройство ввода графической информации, состоящее из основания и курсора (пера), перемещаемого по его поверхности, предназначанное для оцифровки изображений (рис. 13).

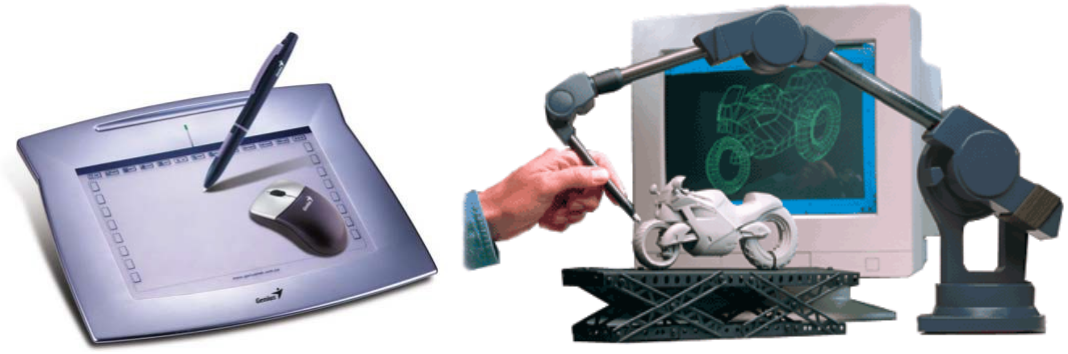


Рис. 13. Дигитайзеры 2D и 3D изображения

Основные области применения дигитайзеров: мультипликация; оцифровывание географических карт для работы с географическими информационными системами; инженерное проектирование, создание прототипов и обратный инжиниринг; научная визуализация.

Основные типы дигитайзеров по принципу работы: ультразвуковые (сонарные), электромагнитные, лазерные, механические.

Печатающие устройства предназначены для вывода результатов обработки информации на бумажный бланкили подобный носитель, т.е. для документального оформления итоговых данных. Бывают принтеры и плоттеры.

К основным техническим характеристикам печатающих устройствотносят: принцип действия; цветовые возможности (черно-белые или цветные принтеры); графические возможности или их отсутствие; разрешающая способность; качество печати, тесно связанное с предыдущим показателем и обобщающее его; скорость печати (быстродействие); стоимость.

Принтер – это устройство, обеспечивающее вывод изображения на бумагу или пленку (рис. 14).

Рис. 14. Лазерный принтер

Классификация выпускаемых принтеров по технологии печати:

1) ударного типа характеризуются тем, что изображение на бумагу наносится механическим способом:

- литерные – производится удар по бумаге литерой (символ) через красящую ленту, в результате чего на бумаге остается контур символа;
- точечно-матричные – печатающая головка перемещается по специальным направляющим вдоль печатаемой на бумаге строки, «вырисовывая» выводимую информацию по точкам через красящую ленту;

2) безударного типа – передвижение бумаги и печатающей головки осуществляется механическим способом, но для формирования изображения на бумаге используются немеханические принципы:

- струйные – изображение наносится на бумагу путем «выстреливания» (под давлением) красителя из крохотного сопла;
- термографические – для нанесения точек используется свойство некоторых материалов изменять свой цвет при нагревании (или расплавляться) тонкими электродами;
- электрофотографические (лазерные) – в процессе работы лазерным диодом в соответствии с выводимым изображением на барабане засвечиваются требуемые участки, и электрический заряд их изменяется, после чего на барабан наносится порошок определенного цвета (тонер), частицы которого обладают заданным электрическим зарядом. Затем рисунок переносится на бумагу путем ее прижатия к барабану и последующего приложения электрического поля и тонер фиксируется на бумаге;
- электростатические – используется принцип ионного осаждения электронная печать);
- электрочувствительные – изображение формируется в результате протекания тока по поверхности специальной бумаги;
- магнитографические – аналогична электрографии и электростатике, но в ней используется магнитная запись.

Плоттер (графопостроитель) – это устройство для автоматического вы-

черчивания с большой точностью на бумаге широкоформатных изображений – рисунков, схем, сложных чертежей, графиков и т. п. – на бумаге размером до А0 или кальке (рис. 15).

Рис. 15. Струйный плоттер

Графопостроители можно классифицировать следующим образом:

- по способу формирования чертежа – с произвольным сканированием и растровые;
- по способу перемещения носителя – планшетные, барабанные и смешанные (фрикционные, с абразивной головкой);
- по используемому инструменту (типу чертежной головки) – перьевые, фотопостроители, со скрайбирующей головкой, с фрезерной головкой.
- по технологии печати – планшетные, перьевые, струйные, электростатические, прямого вывода изображения, на основе термопередачи, лазер- ные, режущие, сольвентные.

Многофункциональное устройство (МФУ), – копировальный аппарат с дополнительными функциями принтера, сканера, факсимильного устройства (рис. 16).

Рис. 16. Многофункциональное устройство

Дополнительные функции могут присутствовать в стандартной комплектации устройства или же добавляться к базовому устройству опционально.

К устройствам хранения данных относятся стримеры, накопители на съем-

ных дисках – дискеты и оптические диски, флеш-диски, *usb*-диски и т.д.

Компьютерный оптический диск – оптический носитель информации, данные с которого считывается с помощью лазера в специальном приводе.

Вид дисков ограничим следующими форматами:

- *CD* – это диск объемом 700 мегабайт (Мб).
- *DVD* – это диск объемом 4,5 гигабайт (Гб).
- *MiniCD* или *MiniDVD* – маленькие диски по размеры, и по объему.
- *Blu-Ray* – это диск объемом 25 гигабайт (Гб) для хранения большого количества информации.

Бывают диски для однократной записи (*CD-R* и *DVD-R*) и перезаписываемые диски для многоразового использования(*CD-RW* и *DVD-RW*). Если на диске стоит знак «+» (плюс) – это значит, что можно дописывать информацию на диск, пока есть свободный объем. Если знак «–» (минус), то больше на данный диск нечего записать нельзя.

Оптические диски могут храниться долго, но они восприимчивы к царапинам, колебаниям температуры и механическим повреждениям.

Стример – устройство долговременной памяти, предназначенное для обеспечения надежного сохранения информации на магнитной ленте. Главное достоинство стримера – возможность достаточно быстро и надежно сохранять большие объемы ценных данных или программ. Поэтому стримеры используются преимущественно для резервного архивирования информации в промышленности, в издательствах, в банках, в деловом и научном мире.

Флэш-диски (*USB Flash Drive*) – это тип внешнего носителя информации для компьютера, устройство для энергонезависимого хранения данных с возможностью многократной перезаписи.

Рис. 17. Подключение флэш-диска

Конструктивно основным элементом хранения информации во флэш-дисках являются микросхемы памяти, выполненные по технологии *Flash*, являясь обычными электрически стираемыми ПЗУ. Эти устройства можно перезаписывать более миллиона раз, емкость этих микросхем достаточно большая при весьма малых габаритах, и время хранения записанной информации в таких микросхемах практически неограничено.

Устройства подключения к каналам связи (телекоммуникационные средства) включают аппаратуру передачи данных, с помощью которой персональная ЭВМ может осуществлять обмен информацией с другими ЭВМ по каналам связи – модемы, сетевые адаптеры, концентраторы, коммутаторы, маршрутизаторы и т.д. Эти устройства необходимы для передачи и приема данных. Телекоммуникационные средства подключают ПК к большим ЭВМ и сетям ЭВМ, в результате обеспечивается доступ пользователей к информационным ресурсам.

Сетевой адаптер служит для подключения к сети некоторого сетевого устройства, например, компьютера или другого сетевого оборудования. Конструкция сетевых адаптеров ориентирована на конкретные методы передачи сетевого сигнала, тип компьютерной шины и сетевую передающую среду.

Модем преобразует выходящий компьютерный (цифровой) сигнал в аналоговый, который может быть передан по телефонной линии, а затем входящий аналоговый сигнал в цифровой, понятный компьютеру.

Роутер или маршрутизатор – это сетевое устройство, которое соединяет различные компьютерные сети и управляет обменом данными между ними. Если необходимо подключить несколько компьютеров, ноутбуков, нетбуков или даже смартфонов к одному интернет-подключению, необходимо построить локальную сеть и установить роутер.

ТЕМА 3. ПРОГРАММНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

План:

1. Программное обеспечение компьютера
2. Базовое программное обеспечение.
3. Системное программное обеспечение компьютера.
   3.1. Операционная система Microsoft Windows.

Вопрос 1. Программное обеспечение компьютера

Программное обеспечение (ПО) – совокупность программ и правил, позволяющая использовать ЭВМ для решения различных задач.

Между программами существует взаимосвязь, многие программы работают, опираясь на программы более низкого уровня, то есть можно говорить о существовании межпрограммного интерфейса. Уровни программного обеспечения представляют собой пирамидальную конструкцию (рис. 18).

Самый низкий уровень программного обеспечения представляет базовое ПО, которое отвечает за взаимодействие с базовыми аппаратными средствами. Базовое ПО входит в состав базового оборудования и хранится в специальной микросхеме, называемой постоянным запоминающим устройством (ПЗУ).

Рис. 18. Уровни программного обеспечения

Программы, работающие на системном уровне, обеспечивают взаимодействие прочих программ компьютерной системы с программами базового уровня и непосредственно с аппаратным обеспечением. Специальные резидентные программы, которые дополняют систему ввода/вывода и обеспечивают обслуживание дополнительных внешних устройств или нестандартное использование имеющихся устройств, называются **драйверами устройств** и входят в состав системного уровня. Другой класс программ системного уровня отвечает за взаимодействие с пользователем, т.е. он получает возможность вводить данные в вычислительную систему, управлять ее работой и получать результат в удобной форме. Совокупность ПО системного уровня образует ядро операционной

системы компьютера.

ПО служебного уровня взаимодействует с программами предыдущих уровней. Основное назначение служебных программ (утилит) состоит в автоматизации работ по проверке, наладке и настройке компьютерной системы, что позволяет расширить или улучшить функции системных программ. В разработке и эксплуатации служебных программ существует два направления: интеграция с операционной системой (изменение свойств системных программ для удобства использования) и автономное функционирование (возможность персональной настойки их взаимодействия с аппаратным и программным обеспечением).

ПО прикладного уровня представляет собой комплекс прикладных программ, с помощью которых на данном рабочем месте выполняются конкретные задания. Спектр этих заданий необычайно широк: от производственных до творческих и развлекательно-обучающих.

Классификация некоторых служебных программных средств:

1. Диспетчеры файлов предназначены для выполнения операций, связанных с обслуживанием файловой структуры: копирование, перемещение переименование файлов, создание каталогов (папок), удаление файлов и каталогов, поиск файлов и навигация в файловой структуре (Проводник, *TotalCommander*, *WinCommander*).

2. Средства сжатия данных (архиваторы) предназначены для создания архивов. Архивирование данных упрощает их хранение за счет того, что большие группы файлов и каталогов сводятся в один архивный файл. При этом повышается эффективность использования носителя за счет повышенной плотности записи информации (*WinRar*).

3. Средства просмотра и воспроизведения используются только для просмотра документов разных типов без возможности их редактирования(*ACDSee, IrfanView,BSPlayer, WindowsMedia, Winamp*).

4. Средства коммуникации позволяют установить соединения с удаленными компьютерами, обслуживают передачу сообщений электронной почты, работу с телеконференциями, и выполняют другие операции в компьютерных сетях.

5. Средства обеспечения компьютерной безопасности классифицируются на средства пассивной и активной защиты данных от повреждения, а также средства защиты от несанкционированного доступа, просмотра и изменения данных. В качестве средств активной защиты применяют антивирусное программное обеспечение (Касперский, *NortonAntivirus, DoctorWeb*).

Классификация некоторых прикладных программных средств:

1. Текстовые редакторы: основные функции этих прикладных программ заключаются во вводе и редактировании текстовых данных. Дополнительные

функции состоят в автоматизации процессов ввода и редактирования (блокнот).

2. Текстовые процессоры: отличие текстовых процессоров от текстовых редакторов в том, что они позволяют не только вводить и редактировать текст, но и форматировать его, т.е. оформлять. К основным средствам относятся – средства обеспечения взаимодействия текста, графики, таблиц и других объектов (*MicrosoftWordPad, Word*)

3. Графические редакторы – программы, предназначенные для создания, редактирования и просмотра графических изображений. Различают:

- растровые редакторы, применяемые, когда графический объект представлен в виде комбинации точек, образующих растр и обладающих свойствами яркости и цвета. Характерно применение для работы с фотографическим и полиграфическим изображением: *Paint* и мощные профессиональные графические системы, например, *Adobe Photoshop* и *CorelPhoto-Paint*;
- векторные редакторы отличаются представлением изображения в виде
- фигур, а не точек. Характерно применение для работы с чертежно-графическими иллюстрациями: графический редактор, встроенный в текстовый редактор *Word,* и профессиональные наиболее распространенные системы *CorelDraw* и *Adobe Illustrator*. Полученные изображения занимают меньше места, но требуют при разработке более высокую производительность компьютера.
- редакторы трехмерной графики используют для создания трехмерных композиций, их также называют 3*D*-аниматорами.

4. Системы управления базами данных. Базами данных называют огромные массивы информации, организованные в табличные структуры. Основными функциями систем управления базами данных являются: создание пустой (незаполненной) структуры базы данных; представление средств ее заполнения или импорта данных из другой базы; обеспечение возможности доступа к данным, их поиска и фильтрации, а в последнее время и защиты (*MicrosoftAccess*).

5. Электронные таблицы представлены средствами для хранения различных типов данных и их обработки, т.е. для преобразования данных в соответствии с их структурой. Основное свойство электронных таблиц состоит в том, что при изменении любых ячеек таблицы может происходить автоматическое изменение содержания во всех прочих ячейках, связанных друг с другом математическими или логическими выражениями (*MicrosoftExcel*).

6. Системы автоматического проектирования (*CAD*-системы) предназначены для автоматизации проектно-конструкторских работ и кроме чертежно-графических работ позволяют проводить простейшие расчеты и выбор готовых конструктивных элементов из собственных баз данных (*Auto-CAD*).

7. *Web*-редакторы объединяют в себе текстовые и графические редакторы

и предназначены для создания так называемых *Web*-документов. *Web*-документы – это электронные документы, при подготовке которых следует учитывать особенности приема/передачи информации в сети Интернет.

8. Браузеры (обозреватели, средства просмотра *Web*), предназначены для просмотра электронных документов, выполненных в формате *HTML* (документы этого формата используются в качестве *Web*-документов). Такие программы воспроизводят не только текст и графику, но и музыку, человеческую речь, видеосюжеты и т.д. (*Opera, InternetExplorer*).

9. Бухгалтерские системы – специализированные системы, сочетающие в себе функции текстовых и табличных редакторов, электронных таблиц и систем управления базами данных и предназначенные для автоматизации подготовки бухгалтерских документов предприятия и их учета.

10. Финансовые аналитические системы используются в банковских и биржевых структурах и позволяют контролировать ситуацию на финансовых, товарных и сырьевых рынках, производить анализ текущих событий, готовить сводки и отчеты.

11. Системы видеомонтажа предназначены для цифровой обработки видеоматериалов, их монтажа, создания видеоэффектов, устранения дефектов, наложения звука, титров и субтитров (*UleadVideoStudio, WindowsMovieMaker*).

Вопрос 2. Базовое программное обеспечение

Самый низкий уровень программного обеспечения представляет базовое программное обеспечение. Оно отвечает за взаимодействие с базовыми аппаратными средствами. Как правило, программные средства этого уровня входят непосредственно в состав базового оборудования и хранятся в специальных микросхемах, называемых постоянными запоминающими устройствами (ПЗУ – *Read Only Memory, ROM*). Программы и данные записываются в микросхемы ПЗУ на этапе производства и не могут быть изменены в процессе эксплуатации. Комплект программ, находящихся в ПЗУ, образует базовую систему ввода-вывода (*BIOS – Basic Input Output System*). Основное назначение *BIOS* на этапе загрузки компьютера – проверить работоспособность системы.

В тех случаях, когда изменение базовых программных средств во время эксплуатации является технически целесообразным, вместо микросхем ПЗУ применяют перепрограммируемые постоянные запоминающие устройства (ППЗУ). В этом случае содержание ППЗУ можно изменять.

BIOS – самый близкий к аппаратуре компонент. Основная функция *BIOS* заключается в управлении стандартными внешними и внутренними устройствами: монитором, клавиатурой, дисководами, принтером, таймером.

Вспомогательные функции реализуются при включении ПК на этапе «загрузки».

1. Автоматическое тестирование аппаратного обеспечения при его включении, в том числе оперативной памяти. Основная часть времени тестирования тратится на проверку ОЗУ (на экране во время тестирования выводятся цифры, которые отражают количество проверенных блоков памяти). В случае обнаружения неисправности, ошибки выводится индикация о найденном сбое оборудования.

2. Поиск сначала на гибком, затем на жестком диске программы-загрузки ОС и загрузка ОС в оперативную память. При этом по окончании тестирования выполняется вызов блока первоначальной загрузки операционной системы с диска. Загрузив в ОЗУ блок первоначальной загрузки, *BIOS* передает ему управление, а он в свою очередь загружает другие модули операционной системы.

3. Инициализация векторов прерывания нижнего уровня (ранжирование устройств ПК по степени значимости, «важности») и обслуживание прерываний.

Основным принципом работы компьютера является принцип, базирующийся на системе прерываний. **Система прерываний** – это совокупность аппаратных и программных средств, обеспечивающих процесс переключения с одной программы на другую и возврат к продолжению прерванной программы за счет операций процессора, называемых прерыванием. **Прерывание** – это операция процессора, состоящая в сохранении состояния процессора, предшествовавшего прерыванию, и установлении нового состояния. Это состояние запоминается в регистрах процессора. Прерывание – это реакция процессора на некоторое условие, возникающее в процессоре или вне его. Реакция выражается в прекращении выполнения текущей команды для обработки возникшего условия. Прерывание иногда позволяет обработать такое условие специальной программой и вернуться к прерванной программе.

Прерывания бывают трех видов:

- аппаратные (например: нажатие клавиши);
- логическое или процессорное (например: деление на ноль);
- программные (например: команды ввода/вывода).

Каждое прерывание имеет уникальный номер от 0 до 255 и с ним связана определенная программа, призванная обслуживать возникшую ситуацию. *BIOS* является «программной оболочкой» вокруг аппаратных средств компьютера, предоставляет возможность другим программам, а также и самой операционной системе обращаться к внешним устройствам компьютера через механизм прерываний. На *BIOS* возложена задача обслуживать прерывания нижнего уровня, т.е. тех, которые требуют непосредственного управления аппаратными компо-

нентами компьютера. Это прерывания с номерами от 0 до 31. Прерывания с номерами от 32 и выше относятся к прерываниям верхнего уровня и обрабатываются модулем обработки прерываний *MSDOS.SYS*.

Вопрос 3. Системное программное обеспечение компьютера

Операционная система (ОС) – комплекс программ, предназначенный для обеспечения определенного уровня эффективности вычислительной системы за счет автоматизированного управления ее работой и представляемого пользователям услуг. Эту систему можно рассматривать как программное продолжение и расширения аппаратуры ЭВМ.

Свойства операционной системы:

1. Надежность. Операционная система должна быть надежна, как и аппаратура на которой работает. Она должна быть в состоянии определить и диагностировать ошибки, а также восстанавливаться после большинства характерных ошибок, произошедших по вине пользователя. Она должна защищать пользователя от его же собственных ошибок или, по крайней мере, минимизировать вред, который он может оказать на все программное окружение, находящиеся в ПК.

2. Защита. Операционная система должна защищать выполняемые задачи от взаимного влияния их друг на друга.

3. Предсказуемость. Операционная система должна отвечать на запросы пользователя предсказуемым образом. Результат выполнения команд пользователя должны быть одним и тем же вне зависимости от последовательности, в которой эти команды посылаются на исполнение (при соблюдении установленных в системе правил).

4. Удобства. Операционная система предлагается пользователю потому, что она намного облегчает его работу и освобождает его от бремени задач по определению различных ресурсов и задач по управлению этими ресурсами. Система должна быть спроектирована с учетом основных факторов человеческой психологии.

5. Эффективность. При распределении ресурсов операционная система должна максимально повысить использование системных ресурсов пользователем. Сама система не должна использовать большое количество ресурсов, так как эти ресурсы становятся недостаточными для удовлетворения запросов пользователя.

6. Гибкость. Системные операции могут настраиваться для согласования поведения пользователя. Ресурсы могут быть увеличены (уменьшены) для того, чтобы улучшить эффективность и доступность.

7. Расширяемость. В процессе эволюции к операционной системе могут быть добавлены новые программные средства.

8. Ясность. Пользователь может оставаться в неведении относительно вещей, существующих ниже уровня интерфейсной системы. В то же время он должен иметь возможность узнать о системе столько, сколько он хочет. В данном случае интерфейсной системой являются правила и функциональные характеристики средств подключения и взаимодействия устройств вычислительной машины.

Современные вычислительные системы состоят из процессоров, памяти, таймеров, дисков, сетевой коммуникационной аппаратуры, принтеров и других устройств. Функцией ОС является распределение процессоров, памяти, устройств и данных между процессами, конкурирующими за эти ресурсы. ОС должна управлять всеми ресурсами вычислительной машины таким образом, чтобы обеспечить максимальную эффективность ее функционирования. Критерием эффективности может быть, например, пропускная способность или реактивность системы. Таким образом, ОС реализует:

- интерфейс пользователя (команды в *MS DOS*, *UNIX*; графический интерфейс в ОС *Windows*);
- разделение аппаратных ресурсов между пользователями (в многопользовательской и многозадачной ОС);
- работу в локальных и глобальных сетях;
- возможность работы с общими данными в режиме коллективного пользования;
- планирование доступа пользователей к общим ресурсам;
- эффективное выполнение операций ввода-вывода;
- восстановление данных и вычислительного процесса в случае ошибок.

Цель управления ресурсами заключается в том, чтобы добиться эффективного использования ресурсов пользователем, а также освободить пользователя от бремени задач по оперированию ресурсами.

Под ресурсами ПК подразумевается следующее:

1. Процессорное время – время доступа к процессору и, следовательно, время счета. Большинство задач при вычислении их на ЭВМ тратит половину времени на ожидание завершения операций ввода/вывода. Экономическая необходимость вынуждает разделять ЭВМ между многими пользователями, одновременно работающими. Таким образом, для эффективного использования процессорного времени требуется сложный механизм разделения времени – механизм, использующий одновременную работу центрального процессора и устройства ввода/вывода информации.

2. Оперативная память. Планирование доступа к оперативной памяти неотъемлемо от доступа к центральному процессору. Программа может выполняться, если есть доступ к центральному процессору, она оказывается в оперативной памяти и исполняется, так как память дефицитна, система должна использовать ее с максимальной эффективностью.

3. Переферийные устройства. С большинством периферийных устройств в каждый момент времени может работать только один пользователь. Такая работа периферийных устройств может привести к неэффективному их использованию, если время счета программы довольно велико. Устройства с быстрым доступом разделяются между пользователями с помощью системы управления файлами. Задержки, возникающие при работе с периферийными устройствами быстрого доступа, вполне удовлетворительны виду скорости этих устройств и в виду интервалов времени между программными запросами ввода/вывода.

4. Ресурсы математического обеспечения – представляют собой доступные пользователю функции, предназначенные для работы с данными и для контроля за выполнением программ. Среди этих ресурсов находятся сервисные программы по управлению файлами и по обслуживанию ввода/вывода, программ системного планирования и системные библиотеки.

Для реализации управления ресурсами разные ОС используют различные алгоритмы, что, в конечном счете, и определяет их облик в целом, включая характеристики производительности, область применения и даже пользовательский интерфейс. Например, алгоритм управления процессором в значительной степени определяет, является ли ОС системой разделения времени, системой пакетной обработки или системой реального времени.

Операционная система является посредником между ЭВМ и пользователем. Операционная система осуществляет анализ запросов пользователя и обеспечивает их выполнение. Запрос представляется последовательностью команд на особом языке директив операционной системы.

Операционная система может выполнять запросы в разных режимах, поэтому операционную систему можно разделить на следующие типы:

1. Операционная система пакетной обработки – это система, которая обрабатывает пакет заданий, т.е. несколько заданий, подготовленных одним или более пользователями. Пакет заданий поступает в ЭВМ и взаимодействие между пользователем и его заданием во время вычислительного процесса невозможно. Данная операционная система может функционировать в однопрограммном и мультипрограммном режимах. Предназначены для решения задач, которые не требуют быстрого получения результатов.

2. Операционная система разделения времени – обеспечивает одновременное обслуживание многих пользователей, позволяет любому пользователю взаимодействовать со своим заданием. Эффект одновременной работы достигается разделением процессорного времени и других ресурсов между несколькими вычислительными процессами, которые заданы разными пользователями. Операционная система выстраивает очередь из поступающих заданий, выделяет квант времени для доступа к центральному процессору каждому заданию со-

гласно очереди. Выполнив первое задание, операционная система 1 отсылает его в конец очереди и переходит ко второму и т. д. В этих системах каждой задаче выделяется небольшой квант процессорного времени, ни одна задача не занимает процессор надолго и время ответа оказывается приемлемым.

3. Операционная система реального времени – это система, которая гарантирует оперативное выполнение запросов в течение заданного интервала времени. Запросы могут поступать от пользователей или от внешних по отношению к ЭВМ устройств, с которыми системы связаны каналами передачи данных. При этом скорость вычислительных процессов в ЭВМ должна согласоваться со скоростью временных процессов, т.е. и ходом реального времени. ЭВМ с данной операционной системой чаще всего работает в однозначном режиме. Необходимое время ответа определяется свойствами объектов (пользователей, внешних устройств), обслуживаемых системой. Применяются для управления различными объектами (такими, как станок, спутник) или технологическими процессами (гальваническая линия, доменный процесс и т.д.) и в информационно-поисковых системах.

4. Диалоговые операционные системы – предназначены для индивидуального пользования и обеспечивают удобную форму диалога ЭВМ с пользователем через дисплей при вводе и выполнении команд. Функционирует операционная система обычно в однопрограммном режиме.

Независимо от типа операционная система чаще всего состоит из относительно компактного ядра – монитора (супервизора) и набора системных программ и данных.

Существует три наиболее распространенных операционных системы для ПК – *Microsoft Windows*, *Apple Mac OS X* и *Linux*. Современные операционные системы используют графический пользовательский интерфейс, позволяя использовать манипулятор, мышь, и управлять указателем на экране, нажимая на различные символы, кнопки, выбирать пункты меню. При этом все это отображается на экране, сопровождаясь графическими и текстовыми сообщениями. До разработки графического пользовательского интерфейса единственным способом управления компьютером была возможность ввода команд в командной строке. Причем пользователь должен был вручную набирать все команды, и результат их выполнения мог узнать только по ответному текстовому сообщению на экране монитора.

 *Microsoft Windows*

В середине 1980-х годов компания *Microsoft* создала операционную систему *Windows*. За эти годы было много различных версий *Windows*, и последние из них – *Windows* 8 (выпущена в 2012), *Windows* 7 (2009) и *Windows Vista* (2007). *Windows* устанавливается практически на все продаваемые новые ПК,

благодаря чему она является самой популярной операционной системой в мире. Перед установкой можно выбрать одну из различных изданий *Windows*, таких как *HomePremium*, *Professional* и *Ultimate*.

 *Apple Mac OS X*

Mac OS представляет собой линейку операционных систем, созданных компанией *Apple*. Она устанавливается на все новые компьютеры *Macintosh* или *Mac*. Последние версии операционной системы *Mac OS X* имеют названия *Mountain Lion* (выпущена в 2012), *Lion* (2011) и *Snow Leopard* (2009). *Apple* также предлагает версию под названием *Mac OS XServer*, которая предназначена для работы на серверах.

 *Linux*

Linux является общедоступной операционной системой с открытым исходным кодом, что позволяет изменять ее, а также распространять и использовать во всем мире. Это сильно отличает ее от операционной системы *Windows*, которая может быть изменена только компанией *Microsoft*. Преимущество *Linux* в том, что она бесплатна. Существует несколько версий *Linux*, отличающиеся дизайном, и самые популярные из них *Ubuntu*, *Mint* и *Fedora*. Большинство серверов в интернете управляется именно операционной системой *Linux* благодаря простоте ее настройки.

У каждой операционной системы графический пользовательский интерфейс имеет свой стиль и оформление, поэтому при смене операционной системы в первое время можно очень долго путаться в меню в поисках необходимого пункта. Но современные операционные системы разрабатывались на основе исследований удобства использования, что максимально приблизило сходство стилей оформления, и сделало их интуитивно понятными.

Кроме локальных существуют также и сетевые ОС. **Сетевая операционная система** обеспечивает доступ к ресурсам удаленных ЭВМ сети и предоставляет ресурсы своей ЭВМ удаленным пользователям. Операционная система компьютерной сети во многом аналогична ОС локального компьютера – она также представляет собой комплекс взаимосвязанных программ, который обеспечивает удобство работы пользователям путем предоставления им некоторого набора услуг. Сеть ЭВМ позволяет пользователю работать со своим компьютером как с локальным и добавляет к этому возможность доступа к информационным и аппаратным ресурсам других ЭВМ сети.

Сетевые средства ОС можно разделить на три компоненты:

1. Средства предоставления локальных ресурсов и услуг в общее пользование – серверная часть операционной системы;

2. Средства запроса доступа к удаленным ресурсам и услугам – клиентская

часть операционной системы;

3. Транспортные средства операционной системы, которые совместно с коммуникационной системой обеспечивают передачу сообщений между компьютерами сети.

Клиентские операционные системы в сетях с выделенными серверами обычно освобождены от серверных функций, что значительно упрощает их организацию. Разработчики клиентских ОС уделяют основное внимание пользовательскому интерфейсу и клиентским частям сетевых служб. Наиболее простые клиентские ОС поддерживают только базовые сетевые службы – обычно файловую службу и службу печати. В то же время существуют так называемые универсальные клиенты, которые поддерживают широкий набор клиентских частей, позволяющих им работать практически со всеми серверами сети.

Многие компании, разрабатывающие сетевые операционные системы, выпускают два варианта одной и той же операционной системы. Один вариант предназначен для работы в качестве серверной ОС, а другой – в качестве клиентской. Эти варианты чаще всего основаны на одном и том же базовом коде, но отличаются набором служб и утилит, а также параметрами конфигурации, некоторые из которых устанавливаются по умолчанию и не поддаются изменению.

3.1 Операционная система Microsoft Windows

Windows – операционная система компании *Microsoft Corp*. Она делает работу на компьютере более комфортной и продуктивной, предоставляет много дополнительных возможностей, поддерживает современное оборудование, содержит эффективные средства диагностики, восстановления и обслуживания.

Операционная система *Windows* включает в себя множество компонентов и обеспечивает пользователям различной квалификации комфортные условия работы. Основная идея, заложенная в основу оболочки *Windows*, – естественность представления информации. Наиболее важными отличительными чертами ее являются следующие:

1. *Windows* представляет собой замкнутую рабочую среду. Практически любые операции, доступные на уровне операционной системы, могут быть выполнены без выхода из *Windows*. Основными понятиями пользовательского интерфейса в операционной системе *Windows* являются окно и *пиктограмма*. Все, что происходит в рамках оболочки *Windows*, в определенном смысле представляет собой либо операцию с пиктограммой, либо операцию с окном (или в окне). Стандартизована в операционной системе *Windows* и структура окон и расположение элементов управления ими. Стандартизованы наборы операций и структура меню для сервисных программ. Стандартны операции, выполняемые

с помощью мыши для всех сервисных и прикладных программ.

2. *Windows* пре*д*ставляет собой графическую оболочку. От пользователя не требуется ввод директив с клавиатуры в виде текстовых строк. Необходимо только внимательно смотреть на экран и выбирать из предлагаемого набора требуемую операцию с помощью манипулятора мышь. Курсор мыши следует позиционировать на поле требуемой директивы меню, или на интересующую пиктограмму, или на поле переключателя систем. Мышь позволяет общаться с компьютером через *Windows* посредством фиксирования курсора мыши, одинарного или двойного щелчка и перетаскивания различных элементов при нажатой кнопке мыши.

В рамках *Windows* пользователь может запустить несколько программ для параллельного (независимого) выполнения. Каждая из выполняемых программ имеет свое собственное окно. Переключение между выполняемыми программами производится несколькими способами.

Работа с объектами предусматривает также и операции с пиктограммами. Пиктограммы можно использовать внутри документов для выполнения функций, подобных тем, которые эти пиктограммы выполняют в среде *Windows*.

Windows – интегрированная программа. Под управлением оболочки *Windows* могут работать не только специальные программы, разработанные для эксплуатации в среде *Windows*, но и «обычные» программы, работающие в среде *DOS*. Оболочка *Windows* обеспечивает эффективный и комфортабельный обмен информацией между отдельными программами, выполняемыми под ее управлением. С понятием интегрированности связывают обычно также возможность совместного использования ресурсов компьютера различными программами.

Работа в о6олочке *Windows* и в *Windows*-приложениях позволяет осуществлять обмен данными между отдельными программами и параллельно выполнять их. Стандартизация интерфейсов отдельных *Windows*-приложений позволяет легко переходить от одного приложения к другому, не начиная каждый раз с нуля.

При запуске *Windows*, согласно настройкам, возможно использование имени и пароля для аутентификации. По окончании загрузки появляется **Рабочий стол** со всевозможными значками и **Панель задач**.

Интерфейс пользователя – набор средств, с помощью которых происходит взаимодействие с операционной системой *Windows*. В это понятие входят окна и пиктограммы, а также все операции, выполняющиеся через позиционирование курсора мыши на пиктограмме или окне.

Интерфейс пользователя русской версии *Windows* полностью реализован на русском языке. Это означает, что система справки, команды меню, а также

встроенные приложения вы будете видеть на русском языке.

Windows поддерживает три типа (стиля) пользовательского интерфейса: *Web*-подобный, классический и смешанный.

При *Web*-подобном пользовательском интерфейсе названия всех открываемых объектов (в частности, папок и файлов) в окнах папок, окне *Проводника Windows* и на рабочем столе подчеркиваются, объекты выделяются одним лишь указанием мышью, а открываются однократным щелчком мыши. Появляется возможность сделать рабочий стол активным. Папки открываются в одном и том же окне.

Классический пользовательский интерфейс более привычен для тех, кто работал в среде *Windows*. Объекты не подчеркиваются, и будут выделяться однократным, а открываться – двойным щелчком мыши. Сделать рабочий стол активным не получится. Каждая папка будет открываться в своем собственном окне.

Смешанный пользовательский интерфейс обеспечивает любую комбинацию свойств, характерных для *Web*-подобного и классического интерфейсов.

Для упрощения работы в своей среде *Windows* предлагает так называемую концепцию рабочего стола. **Рабочий стол** чем-то напоминает поверхность обычного стола. После загрузки *Windows* можно увидеть именно рабочий стол, на поверхности которого располагаются как сами объекты (папки, программы, рисунки, текстовые файлы), так и ярлыки.

Вызов приложений с рабочего стола можно с помощью мыши или заданием горячих клавиш, также по объектам можно перемещаться с помощью клавиатуры.

Прежде чем производить какое-либо действие над объектом *Windows*, его необходимо выделить, т.е. установить курсор мыши на объект и нажать левую кнопку мыши.

Переход к тому или иному объекту *Windows* осуществляется с помощью ярлыков. При двойном клике клавишей мыши по ярлыку открывается то или иное приложение. Удалив ярлык с папки или рабочего стола, потеряется только быстрый доступ к приложению, но не само приложение. Это основное отличие ярлыка *Windows* от пиктограммы. **Пиктограмма** – это небольшая картинка, представляющая на экране компьютера приложение *Windows*. Например, создав ярлык к принтеру и поместив его на рабочий стол, можно просто перетаскивать документы на ярлык принтера и печатать их немедленно. В ярлыке можно указывать, в каком режиме должно загружаться приложение (свернутым, развернутым на весь экран, в стандартном), можно задать рабочий каталог.

Для создания ярлыка необходимо поместить курсор на любое место рабочего стола и щелкнуть правой кнопкой мыши. В появившемся контекстном ме-

ню выбрать команду **Создать**, а следующем меню – **Ярлык**. С помощью кнопки **Обзор** необходимо найти на диске приложение, на которое будет ссылаться ярлык, нажать кнопку **Далее**, ввести имя ярлыка и, наконец, нажать **Готово**. Если надо создать ярлык для документа, то правой клавишей мыши перетащить выбранный документ на рабочий стол или в папку и в появившемся контекстном меню выбрать **Создать ярлык**.

В *Windows* можно удалить ярлык или любой другой объект, просто нажав правую кнопку мыши и выбрав в контекстном меню команду **Удалить**, либо используя клавишу **Delete**.

Объекты (файлы, папки и ярлыки), лежащие на рабочем столе, используются достаточно интенсивно, и от того, насколько удобно они расположены, во многом зависит эффективность работы. Упорядочить эти объекты можно путем сортировки, переразмещения и выравнивания.

Для сортировки объектов необходимо выбрать одну из следующих команд, содержащихся в подменю **Упорядочить значки** контекстного меню фона рабочего стола: **по имени**, **по типу**, **по размеру** или **по дате**.

В любой момент времени имеется возможность включить режим автоматического переразмещения объектов. В результате объекты сгруппируются у левой границы рабочего стола и выровняются, заняв узлы воображаемой сетки. При изменении состава объектов на рабочем столе они будут немедленно переразмещены в соответствии с этим же правилом.

Если режим автоматического переразмещения отключен, предоставляется возможность расположить объекты по своему вкусу, перетаскивая их мышью.

Мой компьютер – папка, в которой целиком представлен персональный компьютер. Здесь содержатся все жесткие и гибкие диски, сетевые диски, устройства для чтения или записи компакт-дисков, личные и сетевые принтеры, личный планировщик заданий и параметры подключения к удаленному компьютеру.

Из папки **Мой компьютер** можно также обратиться к одной из самых важных системных папок *Windows* – **Панели управления**.

Чтобы просмотреть содержимое каталогов и файлов компьютера необходимо открыть папку **Мой компьютер**, дважды щелкнув на соответствующий значок на рабочем столе, а затем выбрать интересующий объект. Чтобы вернуться в предыдущую папку, можно нажать клавишу **Backspace** или кнопку **Назад** на панели инструментов. Просматривать диски и папки на компьютере также можно в окне проводника *Windows*.

Папка **Корзина** обеспечивает дополнительную безопасность при удалении файлов или папок. При удалении файла или папки с жесткого диска *Windows* помещает его в корзину, и значок корзины из пустого становится полным. Элементы, удаляемые с сетевого или внешнего дисков, не попадают в корзину, а

сразу удаляются.

Файлы или папки остаются в корзине до того момента, пока пользователь не удалит их окончательно. Эти элементы все еще занимают место на диске и могут быть восстановлены в их исходное местоположение. Когда корзина заполняется, *Windows* автоматически очищает в корзине необходимое место для наиболее часто удаляемых файлов и папок.

В окне **Мое сетевое окружение** отображаются ярлыки общих компьютеров, принтеров и других сетевых ресурсов. Такой ярлык автоматически создается в папке **Мое сетевое окружение** при открытии какого-либо общего сетевого ресурса, например, принтера или общей папки. Данная папка также содержит гиперссылки на используемые задачи и папки компьютера. С помощью этих ссылок удобно просматривать сетевые подключения, добавлять ярлыки новых мест в сетевом окружении, а также просматривать компьютеры в сетевом домене или в рабочей группе. Если компьютер подключен к рабочей группе, в которой меньше 32 компьютеров, *Windows* автоматически создаст в папке **Мое сетевое окружение** ярлыки общих ресурсов этой рабочей группы.

Панель задач расположена в нижней части экрана *Windows*. Когда открывается приложение или документ, на **Панели задач** появляется соответствующая кнопка, нажав на которую можно перейти в окно этого приложения.

Windows предоставляет в распоряжение пользователя стандартные панели инструментов, которые можно разместить на панели задач или непосредственно на рабочем столе – **Адрес**, **Ссылки**, **Рабочий стол**, **Быстрый запуск** и **Язык**. Они облегчают доступ к часто используемым объектам.

Панель **Адрес** является полем со списком, содержащим адреса часто используемых объектов – папок, документов, приложений и *Web*-страниц. Панель **Адрес** обеспечивает открытие любого объекта путем указания его адреса, который набирается в поле на панели. Единожды введенный адрес запоминается в списке для дальнейшего использования.

Панель **Ссылки** обеспечивает непосредственный доступ к лучшим *Web*-узлам.

Панель **Быстрый запуск** состоит из кнопок, обеспечивающих быстрый запуск выбранных приложений.

Панель **Рабочий стол** содержит набор кнопок, каждая из которых представляет объект, находящийся на рабочем столе. Использование этой панели избавляет от необходимости сворачивать окна приложений, чтобы открыть один из таких объектов.

Панель **Языка** отображает язык, на котором идет набор текстовой информации, и позволяет переключаться с одного языка на другой. Здесь можно установить один или несколько языков, с которыми будет выполняться работа, и

назначить используемый по умолчанию. С помощью этой команды происходит сообщение системе, что язык, который выбран является ключевым языком. *Windows* предлагает на выбор всего два сочетания клавиш, которые могут использоваться для смены языка *LeftAlt + Shift* (*LeftAlt* означает левую клавишу *Alt*) и *Ctrl + Shift*. ОС *Windows* сама активизирует шрифт, соответствующий выбранной раскладке клавиатуры. Кроме того, можно произвести переключение, щелкнув по индикатору языка мышью и выбрав язык в открывшемся меню.

Диалоговое окно **Дата и время** всегда можно вызвать, просто дважды щелкнув мышью в том месте панели задач, в котором отображены часы компьютера. Две вкладки открывшегося диалогового окна позволяют настроить текущую дату и системное время на компьютере.

Открывающая доступ к **Главному меню**, кнопка **Пуск** находится в левой части панели задач и позволяет запустить любое имеющееся на жестком диске или любом другом носителе приложение *Windows*.

Для активизации требуемой команды надо открыть подменю, для чего указать заголовок подменю мышью. Возможно, придется последовательно открыть несколько подменю. Пункты-подменю помечены справа треугольной стрелкой, которая визуально отличает их от пунктов-команд. Для закрытия **Главного меню**, ни одной из команд которого не воспользовались, нужно щелкнуть мышью вне зоны меню.

В главном меню собраны команды:

- запуска практически всех программ, содержащихся в *Windows*, и приложений, установленных на компьютере (**Все программы**),
- поиска локальных и сетевых ресурсов (**Найти программы и файлы**), через который можно найти файлы и папки,
- настройки *Windows* (**Настройка**), которая включает панель управления, принтеры, панель задач,
- завершения работы на компьютере, для выключения персонального компьютера или его перезагрузки.

Многие объекты, которые можно наблюдать на экране монитора при работе в среде *Windows*, имеют **контекстные меню**. К их числу относятся папки и файлы, фрагменты документов, а также элементы пользовательского интерфейса *Windows* – кнопки на всевозможных панелях и сами панели, кнопка **Пуск** на панели задач, фон рабочего стола и открытых папок и т. п. В контекстные меню включаются наиболее распространенные команды, с помощью которых с объектами выполняются те или иные действия. Контекстное меню, как правило, имеется не только у одиночных объектов, но и у группы выделенных объектов. Команда, выданная из такого меню, применяется од-

новременно ко всем объектам. Для выполнения команды контекстного меню, необходимо выделить объект (группу объектов) и щелкнуть по нему правой кнопкой мыши.

Вместе с *Windows* поставляется несколько утилит и приложений, обеспечивающих более удобную работу и эффективное использование ресурсов.

Основные приложения *Windows*:

1) приложения, позволяющие воспроизводить звук, анимацию и видео через проигрыватели компакт-дисков или звуковые адаптеры;

2) сервисные прикладные программы для связи с другими компьютерами и интерактивными службами;

3) служебные прикладные программы, позволяющие обслуживать диски, вести архивацию и сохранять данные и файлы, отображать содержимое буфера обмена, выводить информацию относительно системных ресурсов, вести наблюдение за сервером сети и другими сетевыми подключениями, проводить системные политики на основе сетевых групп, увеличить объем дискового пространства на жестких дисках большой емкости, сжать данные на дисках, наблюдать за производительностью системы:

- **Проводник** – диспетчер файлов, отображающий иерархическую структуру файлов, папок и дисков на компьютере. В нем также отображаются подключенные сетевые диски. С помощью проводника *Windows* можно копировать, перемещать и переименовывать файлы и папки, а также выполнять их поиск. Для того чтобы запустить **Проводник**, надо нажать кнопку **Пуск** и в меню **Все программы** выбрать необходимое;

4) набор программ для работы с Интернет;

5) стандартные программы и дополнительные компоненты системы:

- **Калькулятор** имеет два режима: обычный, предназначенный для простейших вычислений, и инженерный, который обеспечивает доступ ко многим математическим функциям;
- **Блокнот** (*Notepad*) – простой текстовый редактор, предназначенный для создания и редактирования текстовых файлов, не требующих форматирования и не превышающих по размеру 64 Кбайт;
- **Графический редактор** (*Paint*) – графический редактор, используемый для создания простых картин и редактирования изображений в среде *Windows* и включать их в другие приложения, также данный графический редактор можно использовать для просмотра и правки снятых с помощью сканера фотографий и др.

В **Диспетчере задач** отображаются сведения о программах и процессах, выполняемых на компьютере. Кроме того, он служит для отображения ключевых показателей быстродействия компьютера. Для выполняемых программ

можно просмотреть их состояние и завершить программы, переставшие отвечать на запросы. Имеется возможность просмотра активности выполняющихся процессов с использованием до 15 параметров, а также графиков и сведений об использовании центрального процессора и памяти. Кроме того, если имеется подключение к сети, можно просматривать состояние сети и параметры ее работы. Если к компьютеру подключились несколько пользователей, можно увидеть их имена, какие задачи они выполняют, а также отправить им сообщение.

Чтобы открыть **Диспетчер задач** *Windows*, необходимо щелкнуть правой кнопкой на пустом место на панели задач и выберите в контекстном меню соответствующую команду.

Окно **Диспетчера задач** имеет следующие вкладки:

- приложения – отображается состояние выполняющихся на компьютере программ и имеется возможность завершить или запустить программу, а также перейти в окно программы;
- процессы – отображаются сведения о выполняющихся на компьютере процессах (например, допускается отображение сведений об использовании ЦП и памяти, ошибках страницы, счетчике дескрипторов и некоторые другие параметры);
- быстродействие – динамически отображаются следующие сведения о быстродействии компьютера: графики загрузки центрального процессора и использования памяти; общее число дескрипторов, потоков и процессов, выполняющихся на компьютере; общий объем физической памяти, памя- ти ядра и выделения памяти в килобайтах.
- сеть (отображается только для компьютеров с сетевой платой) – возможность просматривать графическое представление сведений о производительности сети; на этой вкладке выводятся сведения о качестве и доступности сетевого подключения, а также о том, имеется ли подключение к одной или нескольким сетям;
- пользователи (доступна, только если на компьютере включено быстрое переключение пользователей и если он входит в рабочую группу или является изолированным и недоступна для компьютеров, которые являются членами сетевого домена) – отображаются пользователи, имеющие доступ к компьютеру, а также состояние сеанса и имена пользователей.

Чтобы завершить программу с помощью **Диспетчера задач** во вкладке **Приложения** надо выбрать программу, которую требуется завершить, и нажать кнопку **Снять задачу**. Все несохраненные данные или изменения в программе будут утеряны.

Механизм обмена данными между приложениями включает использование буфера обмена. Универсальный **буфер обмена** – это некоторая динамически изменяемая область памяти, способная хранить информацию в типовых форма-

тах *Windows*, так как пользователь может записать в него самую различную информацию. Методика работы с буфером обмена одинакова для всех приложений и обычно заключается в том, что пользователь выделяет нужную часть документа или изображения, а затем выбирает команды **Копировать** или **Вырезать**. В первом случае выделенный фрагмент копируется в буфер обмена, во втором – также копируется, но после копирования фрагмент удаляется из документа. Буфер обмена может содержать данные одновременно в нескольких форматах.

Для большей производительности *Windows* помещает много данных в оперативную память компьютера, поэтому, когда возникает необходимость закончить сеанс, всегда завершение работы должно быть корректным с использованием команд меню **Пуск**.

Любое приложение *Windows* использует в своей работе прямоугольную область экрана, называемую **диалоговым окном**. Окна в *Windows* делятся на окна приложений и окна документов. Окно документа всегда содержится в окне приложения, создавшего этот документ.

Несмотря на то, что *Windows* как многозадачная система позволяет одновременно исполнять несколько приложений (каждое приложение – в своем окне), в конкретный момент времени пользователь может непосредственно взаимодействовать только с одним приложением – именно оно воспринимает ввод с клавиатуры. Такое приложение, а также окно, в котором оно исполняется, является активным. Все остальные (хоть и функционирующие) приложения, как и окна, в которых они исполняются, называют пассивными.

Для переключения между приложениями можно использовать следующие способы:

1) найдите на **Панели задач** и нажмите кнопку приложения, которое нужно активизировать;

2) щелчок мышью в зоне окна приложения, которое требуется активизировать;

3) последовательное нажатие клавиши *Tab* при нажатой клавише *Alt* до тех пор, пока требуемое приложение не будет активизировано (использование клавиши *Esc* взамен *Tab* позволяет вместо всех перебрать только те приложения, окна которых не свернуты).

Окно приложения может находиться в одном из трех состояний:

1) нормализованном (нормальном), когда оно занимает часть рабочего стола;

2) развернутом, когда оно занимает весь рабочий стол;

3) свернутом, когда оно представляется только кнопкой на панели задач.

Для перевода окна приложения в требуемое состояние достаточно нажать на одну из кнопок в верхнем правом углу окна. Кроме того, нажатие кнопки приложения на **Панели задач** приводит к восстановлению предыдущего состояния свернутого окна или к сворачиванию несвернутого активного окна, а

двойной щелчок мыши по строке заголовка окна переводит нормальное окно в развернутое состояние, а развернутое – в нормальное.

Как правило, все приложения, предназначенные для разработки документов, имеют систему меню, которая выстраиваются в линейку, располагаемую непосредственно под строкой заголовка окна приложения (содержит информацию об имени соответствующего приложения или диалогового окна). В меню входят команды, которые приложение в состоянии выполнять (а пользователь может выдавать), и, возможно, подменю с подобными командами. Часто используемые команды меню приложения обычно представляются кнопками (пиктограммами) на панелях инструментов, чтобы облегчить доступ к ним.

Щелкнув по выбранному наименованию, можно открыть меню с диалоговыми командами или возможными командами подменю. Для выхода из системы меню после отказа от выдачи команды щелкнуть мышью вне зоны меню.

Передвигая документы относительно окна, можно воспользоваться так называемой полосой прокрутки, которая находится у правого или нижнего края окна. Любая полоса прокрутки снабжена квадратным или прямоугольным бегунком и стрелкой прокрутки.

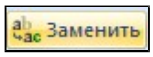
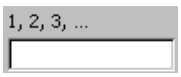
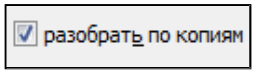
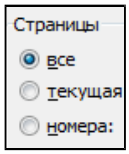
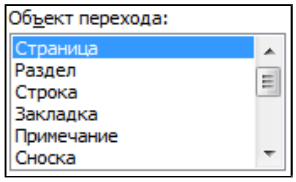
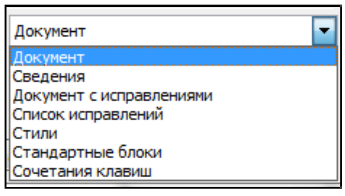
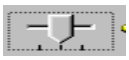
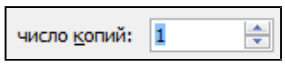
Для создания нового документа через **Кнопку Office**  , находящуюся в верхнем левом углу диалогового окна, необходимо выбрать команду **Создать** и щелкнуть дважды на значке **Новый документ.**

Чтобы загрузить с диска расположенный там файл с рабочим документом, необходимо выполнить команду **Открыть** через **Кнопку Office**  . В результате откроется диалоговое окно загрузки файла, где надо указать диск и выбрать директорию, где расположен файл. Если выбор был сделан правильно, то в левом поле появится список имен файлов, среди которых должен находиться искомый файл. Если щелкнуть по имени этого файла, оно появится в поле **Имя файла.** После этого можно закрыть диалоговое окно, щелкнув мышью по кнопке *Ok* или дважды щелкнув по имени искомого файла.

В каждом диалоговом окне имеется несколько команд для сохранения. При первом сохранении нужно вызвать через Кнопку Office  команду Сохранить как… После этого откроется диалоговое окно, в котором нужно указать имя сохраняемого файла, а также диск и директорию, в которой его надо расположить. Каждое приложение по умолчанию предлагает стандартное имя (например, Doc1.docx или Книга1.xlsx), которое пользователь может заменить любым другим. Предлагаемое по умолчанию программой расширение файла изменять не следует.

Таблица 1

Элементы управления, встречающиеся в диалоговых окнах

Элемент	Назначение и использование
Кнопка 	Обозначает некоторое действие. Чтобы действие выполнить, кнопку нужно нажать – щелкнуть по ней мышью.
Поле ввода 	Позволяет набрать, согласиться с предложенной или отредактировать последовательность символов. Чтобы обеспечить клавиатурный ввод символов в поле, нужно щелкнуть по нему мышью.
Флажок 	Позволяет задать или уточнить режим выполнения команды. Может находиться в двух состояниях – установленном (в квадратике перед ним имеется галочка) или снятом (галочка отсутствует). Смена состояния флажка производится щелчком мыши.
Переключатель 	Представляет собой перечень взаимоисключающих вариантов, из которых выбирается единственный. Эти варианты называются положениями переключателя. Текущее положение отмечается точкой в кружочке.
Поле со списком 	Является комбинацией поля ввода и раскрывающегося списка – видимый элемент можно редактировать или заменять своим элементом, набираемым на клавиатуре.
Прокручиваемый список 	Содержит последовательность элементов, например, имен файлов. В таком списке требуется выделить, по крайней мере, один элемент. Единственный элемент выделяется щелчком мыши.
Раскрывающийся список 	Содержит набор элементов, из которых выбирается единственный. Именно он оказывается видимым. Для выбора другого элемента требуется щелкнуть мышью по кнопке, ограничивающей список справа, а затем – по нужному элементу.
Ползунок 	Позволяет изменять числовое значение связанной с ним величины. Для этого ползунок перетаскивается мышью в требуемое место.
Корешок 	Предназначен для открытия вкладки, которую он представляет. С этой целью по корешку щелкают мышью.
Счетчик 	Является гибридом ползунка и поля ввода. Требуемое значение можно набрать на клавиатуре либо выставить щелчками мыши по кнопкам.

Команда **Сохранить** используется для сохранения изменений, сделанных в существующем документе. Можно также щелкнуть мышью по пиктограмме  , которая находится на основной панели. **Закрыть** и **Выход** используются при закрытии рабочего файла и при завершении работы с приложением соответственно, спрашивается, не хотим ли сохранить изменения.

Простейший способ напечатать текущий документ – выполнить команду **Печать**через кнопку **Office**  , находящуюся в верхнем левом углу диалогового окна. В этом случае будет отображено диалоговое окно печати, в котором можно настроить некоторые параметры печати: выбрать принтер и его свойства, указать количество печатаемых страниц документа, количество копий и другие параметры. Чтобы просмотреть, как будут выглядеть напечатанные страницы, можно выбрать команду **Предварительный просмотр**. Для прекращения печати, пока значок принтера отсутствует в строке состояния, достаточно нажать клавишу *Esc*. Когда значок принтера появился на панели задач, можно дважды щелкнуть по нему и открыть папку принтера. Выделить документ, печать которого надо прекратить, и выбрать команду **Отменить печать** в меню **Документ**.

ТЕМА 4. Офисное программное обеспечение

План:

1. Приемы работы с документами в текстовом редакторе Microsoft Word.
2. Обработка данных средствами электронных таблиц Microsoft Excel.
3. Презентационная графика в Microsoft PowerPoint.
4. Файлы и файловая система.

Вопрос 1. Приемы работы с документами в текстовом редакторе Microsoft Word

Microsoft Word – мощный текстовый редактор, позволяющий быстро создать документ любой сложности. *Word* содержит команды и самые современные средства, такие как встроенная программа проверки правописания и словарь синонимов, которые помогают составлять документы, и готовыми шаблонами, позволяющими сводить воедино заметки, письма, счета и брошюры. *Word* также позволяет работать с документами Интернет.

В *Word* для работы с документами используются меню (панели инструментов); контекстные меню; переключение между документами; справочная система; средства работы с файлами; список шрифтов; возможности автозамены; средства создания таблиц; средства создания формул.

Создание, открытие, сохранение и печать документа выполняется типовым способом для диалоговых окон.

Для копирования, вставки и удаления данных используется вкладка **Главная** панель **Буфер обмена**.

Прежде всего нужно выделить часть документа, которую надо скопировать, а затем выбрать команду  **Копировать**. Необходимо установить указатель мыши на позицию, куда должны быть скопированы данные, и выполнить команду  **Вставить**. Содержимое выделенной области появится в новом месте. В контекстном меню правой кнопки мыши также есть директивы копирования и вставки. Для копирования и вставки можно также использовать комбинации клавиш *Ctrl + C* и *Ctrl + V*.

Если нужно удалить часть документа, то следует использовать директиву **Вырезать**. На панели управления этой директиве соответствует пиктограмма (ножницы)  , а на клавиатуре – комбинация клавиш *Ctrl + X*. Тот же результат достигается просто нажатием клавиши *Delete*.

Действие, которое выполнено последним, можно отменить директивой  **Отменить** или комбинацией клавиш *Ctrl + Z*. Возможно, для поиска действия, которое следует отменить, потребуется прокрутить список. При отмене действия также отменяются все действия, расположенные выше него в списке.

Для настройки параметров полей страницы сразу для всего документа, точку вставки можно поместить в любом месте документа. Чтобы применить установки полей к выделенному тексту, выделите текст до установки полей. При установке параметров полей для выделенного текста *Word* вставит разрыв раздела до выделения и после.

При точной установке размеров полей выделите нужную часть документа и выберите команду **Параметры страницы** во вкладке **Разметка страницы.** В появившемся диалоговом окне выберите вкладку поля. Установите размеры полей в счетчиках **Верхнее**, **Нижнее**, **Левое**, **Правое** и **Переплет**. Выберите переключатель **Положение переплета: слева** или **сверху;** и нажмите кнопку *Ok.*

Параметры полей, возможно также менять при помощи линейки.

Для перемещения по документу используются мышь и клавиатуру. Перемещение при помощи клавиатуры осуществляется стрелками, расположенными на дополнительной панели. Перейти в конец либо начало строки возможно с помощью клавиш *Home* и *End* на дополнительной панели. В конец и начало документа переход осуществляется сочетанием клавиш *Ctrl +Home* и *Ctrl +End.* При желании несложно перейти к определенной странице, сноске и любому другому элементу документа.

Чтобы выделить любой фрагмент текста используйте перемещение курсора мышью при нажатой левой кнопке.

Чтобы выделить слово, дважды щелкните слово.

Чтобы выделить строку текста, переместите указатель к левому краю строки так, чтобы он превратился в стрелку, направленную вправо, после чего щелкните кнопкой мыши. Чтобы выделить несколько строк текста, переместите указатель к левому краю одной из строк так, чтобы он превратился в стрелку, направленную вправо, а затем перетащите указатель вверх или вниз.

Чтобы выделить предложение, удерживая нажатой клавишу *Ctrl,* щелкните предложение.

Чтобы выделить абзац, переместите указатель к левому краю абзаца так, чтобы он превратился в стрелку, направленную вправо, после чего дважды щелкните кнопкой мыши. Другой способ: трижды щелкните абзац. Чтобы выделить несколько абзацев, переместите указатель к левому краю одного из абзацев так, чтобы он превратился в стрелку, направленную вправо, после чего дважды щелкните кнопкой мыши, а затем перетащите указатель вверх или вниз.

Чтобы выделить большой блок текста, щелкните начало фрагмента, прокрутите документ так, чтобы на экране появился конец фрагмента, а затем щелкните его, удерживая нажатой клавишу *Shift.*

Чтобы выделить весь документ, переместите указатель к левому краю текста документа так, чтобы он превратился в стрелку, направленную вправо, после чего трижды щелкните кнопкой мыши.

Чтобы выделить рисунок, щелкните рисунок.

Word предоставляет возможность разорвать страницу в произвольном месте документа, если вставить принудительный разрыв документа. После вставки принудительный разрыв страницы можно удалять, перемещать, копировать и вставлять.

Чтобы вставить разрыв страницы, поместите точку вставки в то место, где должна начаться новая страница, выберите команду **Разрывы** во вкладке **Разметка страницы** на панели **Параметры страницы** и укажите, например, параметр **Следующая страница**.

Для вставки номеров страниц в редактируемый документ выберите команду **Номер страницы** во вкладке **Вставка** на панели **Колонтитулы**. В списке укажите, где следует печатать номера страниц: вверху или внизу страницы, в верхнем или нижнем колонтитуле; также здесь можно выбрать положение номера относительно центра. Через команду **Формат номеров страниц** установите **Формат номера**, и с какого номера начать нумерацию.

Для удаления номеров страниц, выберите команду **Верхний колонтитул**, если номера страниц размещены вверху страницы, или **Нижний колонтитул,** если номера страниц размещены внизу страницы, во вкладке **Вставка** на панели **Колонтитулы**. Далее выделите номер страницы. Если для вставки номеров страниц использовалась команда **Номер страницы** во вкладке **Вставка** на панели **Колонтитулы**, выделите рамку, которая расположена вокруг номера страницы, и нажмите клавишу *Delete*.

Абзац – это фрагмент текста, возможно пустой, который заканчивается маркером конца абзаца. Этот символ вводится путем нажатия клавиши *Enter*. Обычно символ конца абзаца не отображается на экране (для отображения всех невидимых знаков используется пиктограмма ¶ на панели управления **Абзац** во вкладке **Главная**).

Чтобы изменить такие параметры форматирования абзаца, как выравнивание абзацев текста по краям листа, интервалы между строками и абзацами, отступы при помощи команд меню, выделите нужный абзац или абзацы и выберите команду **Абзац** во вкладке **Главная**. В появившемся диалоговом окне выберите параметры форматирования, описанные в табл. 2, и нажмите *Ok*.

Параметры форматирования абзаца возможно, также менять при помощи пиктограмм на панели управления **Абзац** во вкладке **Главная** и линейки.

Минимальным форматируемым в *Word* фрагментом текста является символ. Под форматом символа понимаются параметры: используемый шрифт, его начертания (курсив, полужирный, подчеркнутый), верхний или нижний регистр, цвет шрифта и др. Все эти параметры можно назначать как к отдельному символу, так и ко всему документу. Параметры форматирования можно комбинировать.

Таблица 2

Параметры форматирования абзацев

Параметр	Значение	Описание
Выравнивание	**По левому краю**	Устанавливает выравнивание абзаца по левому краю с неровным правым краем
	По центру	Устанавливает выравнивание абзаца по центру (посередине между полями)
	По правому краю	Устанавливает выравнивание абзаца по правому краю с неровным левым краем
	По ширине	Устанавливает выравнивание абзаца по ширине с равными краями слева и справа
Уровень	**Основной текст, Уровень 1, …, Уровень 9**	Устанавливает уровень текста для создания оглавлений
Отступ	**слева (0 см. по умолчанию или ввести нужную величину)**	Устанавливает отступ от левого поля. Если число положительно, то отступ будет вправо от границы поля; если отрицательно – влево от границы.
	справа (0 см. по умолчанию или ввести нужную величину)	Устанавливает отступ от правого поля. Если число положительно, то отступ будет влево от границы поля; если отрицательно – вправо от его границы.
	Первая строка – нет	Устанавливает отсутствие отступа или выступа.
	Первая строка – Отступ	Устанавливает для первой строки (строк) выделенного абзаца (абзацев) отступ от левого поля (или отступа, если он установлен).
Отступ	**Первая строка – Выступ**	Устанавливает для первой строки (строк) выделенного абзаца (абзацев) выступ от левого поля (или отступа, если он установлен).
	на (1,25 см по умолчанию или ввести нужную величину)	Величина отступа или выступа в сантиметрах влево от левого поля (или отступа, если он установлен).
Интервал	**перед** (0 пт. по умолчанию или ввести нуж ную величину)	Величина интервала в пунктах, на которую выделенный абзац (абзацы) отстоит от предыдущего.
	после (0 пт. по умолчанию или ввести нужную величину)	Величина интервала в пунктах, на которую выделенный абзац (абзацы) отстоит от следующего.
Интервал	**Междустрочный – Одинарный**	Интервал в одну строку. Высота строки подбирается автоматически с учетом размера шрифта, графики и формул, вставленных в строку.
	Междустрочный – Полуторный	Интервал в полторы строки. Вставляет дополнительные полстроки между строками.
	Междустрочный – Двойной	Интервал в две строки. Вставляет дополнительную строку между строками.

	Междустрочный – Минимум	Интервал не меньше, чем указанный в поле значение.
	Междустрочный – Точно	Точный размер интервала указывается в поле **значение.** Некоторые строки могут быть срезаны из-за нехватки места.
	Междустрочный – Множитель	Интервал, равный одинарному с заданным коэффициентом.
	Значение (величина, задаваемая в пунктах или коэффициент умножения)	Межстрочное расстояние. При вводе в это поле какого-либо числа любое значение **Междустрочный** меняется на **Множитель.**

Таблица 3

Параметры форматирования шрифтов

Параметр	Значение	Описание
Шрифт	Список установленных на компьютер шрифтов, а также встроенных шрифтов принтера	Гарнитура шрифта. Обычно присутствуют шрифты *TimesNewRoman, Arial* и *Courier*.
Начертание	**Обычный**	Базовый шрифт. Является основным при наборе.
	Курсив	Текст: *курсив*. Используетсядля выделения.
	Полужирный	Текст: **полужирный.** Используетсядля заголовков и подзаголовков.
	Полужирный Курсив	Комбинация курсива и полужирного.
Размер	**8, 10, 12, 14** и т.д.	Размер символа в пунктах.
Цвет текста	**Авто, Черный, Красный** и т.д.	Изменяет цвет текста.
Подчеркивание	**нет**	Обычный текст.
	одинарное	Обычное подчеркивание. Интервалы между словами подчеркиваются.
	только слова	Обычное подчеркивание. Интервалы между словами не подчеркиваются.
	двойное	Двойное подчеркивание. Интервалы между словами подчеркиваются.
Подчеркивание	**пунктирное**	Подчеркивание с пунктиром. Интервалы между словами подчеркиваются.
	толстой линией	Подчеркивание толстой линией. Интервалы между словами подчеркиваются.
	штриховое	Несколько видов штрихового подчеркивания. Интервалы подчеркиваются.
	штрихпунктирное	Несколько видов подчеркивания штрихпунктирной линией. Интервалы подчеркиваются.
	штрих двойное пунктирное	Несколько видов подчеркивания штрихпунктирной линией с двумя точками. Интервалы подчеркиваются.

	волнистой линией	Несколько видов подчеркивания волнистой линией. Интервалы подчеркиваются.
Видоизменения	**зачеркнутый**	Текст, перечеркнутый линией: ~~зачеркнутый~~**.** Часто используется в режиме исправлений.
	двойное зачеркивание	Текст, перечеркнутый двумя линиями: ~~дважды~~ зачеркнутый.
	верхний индекс	Текст, приподнятый над базовой линией. – (F^x)
	нижний индекс	Текст, опущенный ниже базовой линии. – (F_x)
	с тенью	Текст с тенью справа и снизу: с тенью
	контур	Отображается граница каждого символа: контур
Видоизменения	**приподнятый**	Эффект выпуклости букв: приподнятый
	утопленный	Эффект вдавленности букв: утопленный
	малые прописные	Используется для графического эффекта или специального выделения: МАЛЫЕ ПРОПИСНЫЕ.
	все прописные	Используется для графического эффекта или выделения: ВСЕ ПРОПИСНЫЕ.
	скрытый	Текст не появляется на экране и не печатается, пока это не нужно. Часто используется для личных пометок или комментариев.

Чтобы изменить параметры форматирования символов, выделите нужный фрагмент текста и выберите команду **Шрифт** во вкладке **Главная**. В появившемся диалоговом окне выберите параметры форматирования, описанные в табл. 3, и нажмите *Ok*.

Параметры форматирования абзаца возможно также менять при помощи пиктограмм на панели управления **Шрифт** во вкладке **Главная.**

Для придания документу профессионального вида используется расстановка переносов. Таким образом, можно уменьшить пустые области при выравнивании по ширине или выровнять строки текста в узких колонках. *Word* позволяет расставлять переносы в тексте как автоматически, так и принудительно. Чтобы осуществить расстановку переносов, выполните команду **Расстановка переносов** на панели управления **Параметры страницы** во вкладке **Разметка страницы.** Из представленных вариантов выберите режим **Авто.** Для установки ширина зоны переноса слов укажите интервал, который следует оставлять между концом последнего слова строки и правым полем страницы, с использованием команды **Параметров расстановки переносов**. Чтобы уменьшить количество переносов, сделайте зону переноса более широкой. Чтобы уменьшить неровность края правого поля страницы, сделайте зону переноса более узкой. В поле **Максимальное число последовательных переносов** укажите максимальное число идущих подряд строк, которые могут заканчиваться переносами.

Существует возможность поиска и замены текста, форматирования и специальных символов: символов абзаца, полей, рисунков и т. п. Для этого выберите команду **Найти** или **Заменить** на панели **Редактирование** во вкладке **Главная**. Если в диалоговом окне не видна кнопка **Специальный**, нажмите кнопку **Больше**. Через кнопку **Специальный** выберите нужный элемент. Кроме того, можно ввести код элемента непосредственно в поле **Найти**. Если указанный элемент требуется заменить, введите то, чем его следует заменить, в поле **Заменить на**. Нажмите кнопку **Найти далее**, **Заменить** или **Заменить все**.

При поиске и замене текста в документах можно использовать подстановочные знаки. Например, знак «?» найдет один любой символ, а сочетание «к?т» – «кот» и «кит».

Примечание. Чтобы прервать поиск, нажмите клавишу *Esc*.

Для создания колонок из всего текста документа выберите команду **Выделить все** на панели **Редактирование** во вкладке **Главная** или комбинацией клавиш *Ctrl + A*; из части текста документа – требуемую часть; из раздела документа – установите курсор в текст этого раздела. Чтобы осуществить разбиение документа на колонки, используется команда **Колонки** на панели **Параметры страницы** во вкладке **Разметка страницы**. Здесь возможно выбрать количество колонок, их ширину и промежутки между ними. Изменить ширину колонок и промежутки между ними можно также путем перетаскивания соответствующих маркеров на горизонтальной линейке.

Списки – это фрагменты текста, пункты которого отмечены специальными знаками. Списки могут быть маркированными, нумерованными и многоуровневыми.

Для работы со списками служат кнопки панели **Абзац** во вкладке **Главная**.

Список можно создавать изначально, а можно из уже существующего текста. Если необходимо сделать список из уже существующего документа, то надо выделить фрагмент текста, который подлежит форматированию и выбрать тип списка. При этом выделенный текст будет разбит по пунктам списка согласно абзацам (каждый абзац – это новый пункт списка). Во время выбора типа списка при наведении курсора на соответствующий вариант выделенный текст будет сразу предварительно форматироваться, давая оценить пригодность того или иного варианта.

При формировании многоуровневого списка, чтобы задать создание маркеров очередного уровня можно использовать клавишу *Tab* (либо кнопку **Увеличить отступ** на панели **Абзац** во вкладке **Главная**). Вернуться к вводу данных предыдущего уровня можно, нажав сочетание *Shift + Tab* (либо кнопку **Уменьшить отступ** на панели **Абзац** во вкладке **Главная**).

При работе с маркированными и нумерованными списками можно создавать свой стиль оформления. Для этого нужно в соответствующих диалоговых

окнах выбрать пункт **Определить новый маркер** или **Определить новый формат номера**.

Иногда бывает необходимо в нумерованном списке начать список не с первого номера. Для этой цели служит пункт **Задать начальное значение**. В появившемся окне в зависимости от поставленной задачи надо установить переключатель в одно из двух положений: **Начать новый список** или **Продолжить предыдущий список**. В поле **Начальное значение** необходимо задать номер первого пункта списка.

Word автоматически создает новый нумерованный список, когда абзац начинается с цифры 1 с точкой.

Математическая формула может быть вставлена в любое место документа. Приложение, используемое для создания и редактирования формул, называется **редактор формул.**

Редактор формул позволяет создавать сложные формулы, выбирая символы с панели инструментов и вводя переменные и числа. При создании формул размер шрифтов, интервалы и форматы автоматически регулируются для сохранения соответствия математических типов. Изменять форматирование можно и в процессе работы. Существует также возможность переопределять автоматические стили.

Для вставки формулы в документ надо поместить курсор в то место, где должна появиться формула и выбрать во вкладке **Вставка** на панели **Символы** команду **Формула**. При этом автоматически появляется поле **Место для формулы** и открывается вкладка **Конструктор**, на которой отображаются необходимые для ввода элементы.

На данной вкладке редактора формул расположены кнопки сервисов, символов и структур для вставки в формулу. Для оформления формул можно использовать кнопки, предназначенные для вставки шаблонов или структур, включающих символы типа дробей, радикалов, сумм, интегралов, произведений, матриц или различных скобок или соответствующие пары символов типа круглых и квадратных скобок. Многие шаблоны содержат специальные поля, предназначенные для ввода текста и вставки символов. Шаблоны можно вкладывать один в другой для построения многоступенчатых формул. Для вставки соответствующего элемента в формулу нажмите кнопку панели инструментов.

В *MS Word* есть уже несколько предустановленных формул, одну из которых можно выбрать в списке **Формула** в левой части вкладки **Конструктор**.

Если часто приходится использовать одну и ту же формулу, то можно ее сохранить, и она будет отображаться в блоке встроенных формул. Для этого надо нажать на стрелочку в нижнем правом углу области ввода формулы и выбрать пункт **Сохранить как новую формулу…**

Для форматирования элементов формулы можно выделить их, перейти во

вкладку **Главная** и использовать все доступные функции панели **Шрифт**.

Для изменения элементов формулы необходимо установить указатель на нужную часть формулы, при этом откроется вкладка **Конструктор**. Можно добавить, изменить или удалить элементы формулы. Также можно изменить размер, стиль или форматирование текста либо изменить интервалы и расположение элементов.

После окончания работы нужно установить указатель вне окна формулы и нажать кнопку мыши для возвращения в документ.

При вводе текста в формулу используется команда **Обычный текст** во вкладке **Конструктор** на панели **Сервис**. К символам, оформленным этим стилем, не применяется специальное форматирование, которое автоматически применяется ко всем остальным элементам формулы. Как правило, данный стиль целесообразно использовать для ввода в формулы поясняющего текста, например, «для всех».

Для удаления элементов формулы используются те же действия, что и для обычного текста. Для удаления самой формулы необходимо выделить поле, в котором она вводилась, и нажать *Delete*.

Таблицы используются для упорядочивания данных и создания интересных макетов страницы с последовательно расположенными столбцами текста или графики. Наиболее быстрый путь создания простой таблицы – перейти во вкладку **Вставка** на панель **Таблицы** в меню **Таблица** и в открывшемся окне выбрать необходимое количество строк и столбцов, указав на ячейки мышью. Либо используются команды **Вставить таблицу** или **Нарисовать таблицу** на этой же панели.

В документах *Word* все таблицы по умолчанию имеют тонкую черную линию границы, которая выводится при печати. *Word* не печатает метки символа ячейки или символа строки.

После вставки таблицы автоматически появляются дополнительные вкладки **Конструктор** и **Макет**, где собраны команды редактирования и изменения таблиц.

С помощью команды **Нарисовать таблицу** можно легко создать более сложную таблицу, например, такую, которая содержит ячейки разной высоты или различное количество столбцов на строку – метод, сходный с рисованием таблицы от руки.

Можно создать новую чистую таблицу и заполнить пустые ячейки, либо можно преобразовать существующие абзацы текста (разделенные, например, символами табуляции) в таблицу. Также таблица создается на основе существующего источника данных, такого как база данных или электронная таблица.

Для преобразования текста в таблицу нужно указать места разделения тек-

ста на столбцы и строки, вставив в соответствующие места символы-разделители. Например, вставка символов табуляции для разделения столбцов и вставка отметок абзаца для выделения концов строк. Затем выбрать текст для преобразования и выполнить команду **Преобразовать в таблицу** в меню **Таблица** на панели **Таблицы** во вкладке **Вставка**.

При преобразовании таблицы в текст можно определить запятые, символы табуляции, отметки абзацев или другие символы в качестве разделителя преобразуемого текста. Выберите нужные строки таблицы для преобразования их в абзацы. Затем выполните команду **Преобразовать в текст** на панели **Данные** во вкладке **Макет.** В группе **Разделитель** выберите символ, который будет использоваться на границах столбцов. Строки разделяются отметками абзацев.

При добавлении строк и столбцов в таблицу необходимо указать ячейку, выше или ниже которой будут вставляться новые строки, и левее или правее которой будут вставляться новые столбцы. Для вставки используются соответственно команды **Вставить сверху**, **Вставить снизу**, **Вставить слева** и **Вставить справа** на панели **Строки и столбцы** во вкладке **Макет**. Для добавления нескольких строк или столбцов можно до выбора соответствующей команды выделить нужное их количество для вставки. Можно также использовать инструмент **Нарисовать таблицу** для прорисовки строки или столбца в нужном месте. Чтобы добавить строку в конец таблицы, нужно указать последнюю ячейку последней строки, а затем нажать *Tab*.

Для преобразования группы ячеек в одну ячейку необходимо выделить преобразуемую группу ячеек, а затем выбрать команду **Объединить ячейки** на панели **Объединить** во вкладке **Макет.** Чтобы разбить ячейку на несколько ячеек, надо указать ячейку, подлежащую разбиению, далее выбрать команду **Разбить ячейки** на панели **Объединить.** В результате появится диалоговое окно, в полях **Число столбцов** и **Число строк** которого заноситься число столбцов и строк для разбиения ячейки.

Можно изменить ширину столбцов или высоту строк (или всей таблицы) при помощи перетаскивания границ столбца или строки в самой таблице или перетаскиванием маркеров границ столбцов или строк таблицы на горизонтальной или вертикальной линейке. Удерживание нажатой клавиши *Alt* при перетаскивании маркеров на линейке заставит *Word* отобразить измерения ширины. Если требуется задать конкретное значение ширины столбца или высоту строки, можно указать численные значения в соответствующих полях на панели Р**азмер ячейки** во вкладке **Макет** либо воспользоваться окном **Свойства таблицы,** открыв его используя команду **Свойства** на панели **Таблица** во вкладке **Макет**.

В документе *Word* можно добавлять границу с любой стороны таблицы,

отдельной ячейки или группы ячеек, абзаца или просто выделенного текста. Для создания фона таблицы, абзаца или выделенного текста можно использовать заливку или штриховку. Соответствующие команды редактирования находятся на панели **Стили таблицы** во вкладке **Конструктор.**

Для добавления или удаления границ таблицы выделите требуемые ячейки и выберите команду **Границы и заливка** в меню **Формат** (вкладка **Граница**). Или же выберите команду **Свойства таблицы** в меню **Таблица,** в открывшемся диалоговом окне перейдите во вкладку **Таблица** и нажмите кнопку **Границы и заливка**.

Для удаления из документа ненужной таблицы, необходимо выделить ее и выполнить команду **Удалить таблицу** в меню **Удалить** с панели **Строки и столбцы** во вкладке **Макет**. При удалении ячеек, строк или столбцов используются команды **Удалить ячейки, Удалить строки** или **Удалить столбцы** в этом же меню. Для удаления содержимого таблицы нужно выделить ее и нажать кнопку *Delete*.

Рисунки в документ *Word* можно вставить как объекты из внешних файлов или создать собственные непосредственно графическими возможностями *Word*.

Чтобы создать рисунок *Word* надо обратиться к панели инструментов **Иллюстрации** во вкладке **Вставка.** На этой панели расположены инструменты для рисования, управления и форматирования всех видов графических объектов. Прежде чем нанести один из объектов рисунка, желательно вставить область, в которой они будут располагаться, для удобства дальнейшего редактирования. Выполнить данное действие можно с помощью команды **Новое полотно** из меню **Фигуры** на панели **Иллюстрации** во вкладке **Вставка.** После вставки любого объекта с панели **Иллюстрации** автоматически открывается дополнительная для редактирования рисунков вкладка **Формат**.

Word поставляется вместе с набором готовых фигур, предназначенных для использования в документах. Меню **Фигуры** на панели **Иллюстрации** содержит несколько категорий фигур: линии, основные фигуры, фигурные стрелки, блок-схемы, звезды, ленты и выноски. Для фигур допускаются следующие действия: изменение размеров и цвета фигуры, вращение, отражение, а также комбинирование с другими фигурами, например, кругами и квадратами для составления более сложных фигур. При изменении наиболее выступающих деталей фигуры, используются маркеры изменения формы.

Для добавления надписи в рисунок *Word* используйте кнопку команду **Надпись** на панели **Текст** во вкладке **Вставка**. Если перед созданием надписи не был выделен фрагмент текста или какой-либо объект, то будет создана пустая надпись. В противном случае выделенный фрагмент будет автоматически перенесен в надпись. Чтобы добавить текст надписи, надо ввести его, как в ос-

новном тексте документа. Для добавления или изменения текста в существующей надписи необходимо установить точку вставки в то место надписи, где нужно добавить или отредактировать текст, а затем ввести новый текст и для возврата в основной документ щелкнуть в любом месте документа вне надписи. Чтобы удалить надпись вместе с ее содержимым, выделяется контур надписи с помощью мыши, и используется кнопка *Delete*.

Для удаления графического объекта надо его выделить и нажать клавишу *Delete* или кнопку **Вырезать**.

Кроме возможности выполнять рисунки в *MS Word* появилась возможность создавать диаграммы, таким образом устанавливая взаимосвязь с электронными таблицами *MS Excel*. Для начала построения диаграммы необходимо нажать кнопку **Диаграмма** на панели **Иллюстрации** вкладки **Вставка**, после чего выполняются шаги в мастере построения диграмм *MS Excel*.

Вопрос 2. Обработка данных средствами электронных таблиц Microsoft Excel

Excel – это составная часть *Microsoft Office*, предназначенная для создания электронных таблиц. Самым большим преимуществом Excel является возможность прослеживать, анализировать и выполнять вычисления с данными. Интерфейс программы схож с *Microsoft Word*.

При запуске *Excel* открывается рабочая книга (с первоначальным именем **Книга1**) с пустым рабочим листом. Можно одновременно иметь несколько открытых книг и позднее сохранить их под разными именами. На имена файлов распространяются все правила *Windows*. Тип документа .xls или .xlsx. Если папка не указывается, то файл сохраняется в текущей папке. Ее имя показано при сохранении **Сохранить как.**

Непосредственно под строкой заголовка окна находятся вкладки, содержащие разнообразные команды. Ниже располагается строка формул.

Данные в *Excel* хранятся на рабочих листах. В новой рабочей книге имеется три рабочих листа (с названиями **Лист1, Лист2, Лист3**). Эти названия указаны на ярлыках листов в нижней части экрана. Чтобы изменить заголовок листа, надо выделить его и набрать новое название.

Другие листы можно добавить с помощью специального маркера, расположенного за последним листом. Чтобы перейти к листам, которые не отображаются, используются кнопки перехода, которые находятся перед первым листом.

Перемещение, копирование, удаление можно производить одновременно для нескольких листов. Для этого их необходимо выделить. Для выделения не-

скольких листов щелкают по ярлычку каждого листа, удерживая нажатой клавишу *Ctrl*. Чтобы выделить несколько листов, расположенных рядом, достаточно щелкнуть по ярлычку первого и последнего листа, удерживая нажатой клавишу *Shift*. Чтобы выделить все листы, необходимо выбрать из контекстного меню любого ярлычка листа команду **Выделить все листы**.

Рабочий лист состоит из строк (имена строк заданы цифрами) и столбцов (имена столбцов заданы буквами латинского алфавита *A, B, C*…). Данные в *Excel* хранятся в ячейках на пересечении строк и столбцов. Каждая ячейка имеет название, соответствующее заголовку столбца и строки, называемое адресом или ссылкой (*A*1, *B*1, *C*3…). На пересечении столбца из номеров строк и строки из названий столбцов находится кнопка **Выделить все**.

Создание, открытие, сохранение и печать документа выполняется типовым способом для диалоговых окон.

Прежде чем работать с ячейками, надо выделить одну из них или целую группу. Если выделена одна ячейка, то она становится активной, и ее адрес появляется в поле имени в левой части строки формул. Часто приходится выделять группу ячеек, называемых **диапазонами.**

Для выделения одной ячейки надо щелкнуть на ней левой кнопкой мыши. Для выделения диапазона ячеек надо перетащить указатель мыши по этим ячейкам или указать первую ячейку и ***Shift* + последняя ячейка**. Чтобы добавить диапазон к выделенному диапазону, необходимо перед выделением следующего диапазона нажать клавишу ***Ctrl***. Для выделения столбца или строки используют их заголовки.

В ячейки рабочего листа можно ввести два типа данных: константы и формулы. Константы делятся на числовые значения, текстовые значения и значения дат и времени.

Чтобы ввести числовое значение, надо выделить ячейку и ввести с клавиатуры число. Вводимые цифры отображаются в строке формул и в активной ячейке. Мигающая вертикальная черта, которая появляется в строке формул и в активной ячейке, называется точкой вставки. По окончании ввода значение надо зафиксировать нажатием клавиши **Enter.** Обычно после этого активизируется ячейка на одну строку ниже текущей ячейки.

Специальные символы:

1) если ввод числа начинается со знака «+» или «–», то *Excel* опускает «+» и сохраняет «–»;

2) символ **E** или **e** используется при вводе чисел в экспоненциальном представлении (1e2 = 100);

3) числовые значения, заключенные в круглые скобки принимаются за отрицательные (-3 = (3));

4) для отделения десятичных знаков от целых используются точка или запятая;

5) если ввод заканчивается знаком **%**, то к ячейке применится процентный формат;

Вводимый текст, который не может быть полностью отображен в одной ячейке, перекрывает соседние ячейки, но находится при этом в активной.

Можно настроить ширину столбца, установив указатель мыши в области заголовков столбцов на линии, отделяющей нужный столбец от столбца справа. Когда указатель примет вид перекрестия с двусторонней стрелкой, перетащить линию, разделяющую столбцы, при нажатой левой кнопке мыши, или, используя вкладку **Главная** панель **Ячейки** меню **Формат** команду **Ширина столбца**, набрать нужную ширину столбца. Высоту строки настраивают, установив указатель мыши в области заголовков строк на линии под номером строки, когда указатель примет вид перекрестия с двусторонней стрелкой, перетащить линию, разделяющую строки, при нажатой левой кнопке мыши, или, используя вкладку **Главная** панель **Ячейки** меню **Формат** команду **Высота строки**, набрать нужную высоту строки.

Перенос текста в ячейке можно осуществить, используя флажок **Переносить по словам** панели **Выравнивание** вкладки **Главная**.

Для объединения ячеек нужно выделить ячейки, которые надо объединить, и установить флажок **Объединение ячеек** панели **Выравнивание** вкладки **Главная**, либо воспользоваться специальной кнопкой  на панели **Выравнивание**.

Для ввода дат и времени в ячейках устанавливается формат **Дата** или **Время.** Также дату можно ввести, если набрать число через дробь (1/2 = 01.фев).

К данным хранящимся в ячейках можно применять различные форматы, т.е. изменять представление данных, не меняя их смыслового содержания.

Для текстовых данных в этом случае используется панель **Шрифт** во вкладке **Главная**. Для числовых данных используются форматы, которые можно выбрать во вкладке **Главная** панель **Число**:

1. **Общий** устанавливается по умолчанию за несколькими исключениями. Этот формат отображает точно то, что ввели в ячейку. Например, 123,45 в ячейке будет 123,45. Общий формат не выводит незначащие нули, вместо 123,0 выведет 123.

Исключения: Длинные числовые значения отображаются в экспоненциальной записи или округляются. Общий формат выведет целое число 12345678901234, как 1,23457E+13. Если ввести значение 123456,781234 в ячейку со стандартной шириной и применить формат **Общий**, то будет выведено число 123456,8.

2. **Числовой** имеет параметры, позволяющие выводить числовые значения в виде целых чисел или чисел с фиксированной запятой, а так же выделять отрицательные значения с помощью цвета. При установке этого формата надо выбрать:

- **число выводимых десятичных знаков,** вводя или прокручивая значение в соответствующем поле;
- **разделитель групп разрядов** добавляет пробелы между сотнями и тысячами, тысячами и миллионами;
- **отрицательные числа** позволяет изменять формат вывода отрицательных чисел и выводить их красным цветом.

3. **Денежный** аналогичен формату в категории числовой, но вместо разделителя групп разрядов появляется возможность управлять выводом символа денежной единицы.

4. **Финансовый** выводит денежную единицу с выравниванием по левому краю, а само число выравнивается по правому краю; тире вместо нулевых значений. Для удобства применения данного формата можно выбрать соответствующую кнопку  на панели **Число**.

5. **Процентный** формат выводит числа в виде процентов. Десятичная запятая в формируемом числе сдвигается на два знака вправо, а знак процента выводится в конце числа. Например, процентный формат без десятичных знаков выведет число 0,1234 как 12%, а при задании двух десятичных знаков это значение будет выведено как 12,34%.

Процентный формат без десятичных знаков можно применить, нажав соответствующую кнопку  на панели **Число**.

6. **Дробный** выводит дробные значения как фактические дроби, а не в виде десятичных значений. Если ввести 123,456 в ячейку, к которой применен формат **Шестнадцатыми долями**, *Excel* выведет 123 7/16.

7. **Экспоненциальные** используется для вывода очень больших и очень малых чисел.0,000000009 будет выведено как 9,00E-09.

8. **Текстовый** – значение в этой ячейке должно трактоваться, как текст. Применение к формам текстового формата обычно используется как способ исключения формулы из листа без фактического ее удаления. Можно отформатировать формулу как текст и затем найти другие зависимые формулы, которые возвращают ошибочные значения.

9. **Дополнительные** форматы используются для написания почтового индекса, номера телефона, табельного номера и др.

Чтобы удалить назначенные форматы, надо выбрать вкладку **Главная** панель **Редактирование** команду **Очистить форматы.** Чтобы удалить значения в ячейках, выбрать команду **Очистить все**.

Нельзя забывать разницу между хранимыми и отображаемыми значениями. На хранимые числовые или текстовые значения в ячейках форматы не воздействуют.

Все формулы, которые создаются в *Excel*, должны начинаться со знака равенства (=), который говорит, что следующие символы образуют формулу. Значения, которые появляются в ячейке, являются отображаемыми значениями (результаты), а в строке формул выводятся хранимые значения (как получается результат). Например, в ячейке А1 вводим = 10 + 5 *Enter*, то в ней появится число 15.

В *Excel* принят следующий приоритет операций:

1) в первую очередь вычисляются выражения в круглых скобках;

2) умножение, деление, сложение, вычитание;

3) операции с одинаковым приоритетом выполняются слева направо.

Если в формуле количество открывающихся и закрывающихся скобок не совпадает, то выдается сообщение об ошибке.

Ссылка является индикатором ячейки или диапазона ячеек в книге. Создавая формулу, содержащую ссылки на ячейки, мы связываем формулу с ячейками книги. Например, в ячейку А1 вводим = 10*2, в ячейку А2 вводим = А1, в ячейку А3 вводим = А1 + А2, и в ней получим 40.

В формулах ссылки на ячейки можно вводить с помощью мыши, не набирая с клавиатуры. При щелчке на ячейке вокруг нее появляется подвижная рамка, и ссылка на эту ячейку вставляется в формулу. По окончании ввода надо нажать *Enter***.**

Ссылки бывают абсолютные, относительные и смешанные.

Адреса, которые автоматически меняются при копировании их в новые ячейки, называются относительными адресами. При копировании формулы, содержащей относительный адрес, эти адреса изменяются в соответствии с новым положением формулы, и записывается так **A1.**

В определенных случаях требуется, чтобы некоторые ссылки на ячейки не изменялись при перемещении формулы. Для того, чтобы запретить изменение адреса ячейки при копировании формулы на новое место, необходимо сделать адрес этой ячейки абсолютной ссылкой. Абсолютные ссылки всегда относятся к одним и тем же ячейкам независимо от того, где находится формула. Абсолютная ссылка задается с помощью знака $ перед заголовком строки и столбца и записывается так **A1.**

Смешанная ссылка содержит относительную и абсолютную ссылку и записывается либо **A$1**, либо **$A1.**

Ссылки на другие листы в той же книги записываются следующим образом для ячейки В10 листа 2 в ячейку А4 листа 1:

1) выделить ячейку А4 листа 1 и ввести знак равенства;

2) перейти на лист 2 и выделить ячейку В10, нажать Enter;

3) после этого будет активен лист 1, и в строке формул появиться формула **= Лист2!B10**.

Ссылки на листы в другой книге называются внешними и записываются следующим образом **= [Книга2]Лист1!B1** (часть ссылки, указывающая на книгу, заключена в квадратные скобки, а ссылка на ячейку является абсолютной). Если книга, на которую указывает ссылка, закрыта, то появляется полный путь к папке, где храниться эта книга **=’C:\Excel\[Книга2]Лист1!’B1** (часть ссылки, указывающая на книгу и лист, заключена в кавычки).

Иногда копирование формул существенно проще создания новых. Для копирования необходимо выполнить следующие действия:

- выделить ячейку, в которой расположена формула для вычисления;
- поместить указатель мыши в правом нижнем углу ячейки так, чтобы он принял форму черного крестика «+»;
- перетащить маркер заполнения на следующую ячейку;
- после того как кнопка мыши будет отпущена, в заполняемой ячейке появится копируемая формула.

После копирования формулы будет отображена кнопка **Параметры автозаполнения**.

Примечание: маркер заполнения может быть использован для копирования формул только в соседние ячейки по горизонтали или вертикали.

С текстом в формулах можно выполнять числовые операции, если текстовые значения содержат следующие символы 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, e, /, $, %, ( ), но текст должен быть заключен в кавычки.

Например, выражение в строке формул = “$123” + “$12” даст результат, в ячейке получим 145 без знака $, или при вводе ячейку зависимости: “да”, если А1 > 2, в обратном случае “нет”, формула выглядит следующим образом: ЕСЛИ(А1 > 2; “да”;“нет”).

В *Excel* можно группировать ячейки в диапазоны данных. Диапазоны можно выделить либо с помощью мыши, либо с помощью клавиатуры. Можно комбинировать диапазоны ячеек в формулах при вычислениях.

Операторы ссылок позволяют определить, как группировать ячейки или диапазоны в вычислениях (табл. 4).

Например, при вычислении формула

- СУММ(В12:В15) суммирует данные в ячейках В12, В13, В14, В15;
- СУММ(В12:В15;С5) суммирует данные в ячейках В12, В13, В14, В15, С5;
- СУММ(А1:А5 А4:А9) суммирует данные в ячейках, входящих в оба диапазона, т.е. А4, А5.

Таблица 4

Использование специальных знаков в формулах

оператор	описание
: двоеточие	Определяет диапазон, включающий две указанные ячейки и все ячейки между ними.
; точка с запятой	Определяет объединение, комбинацию нескольких ссылок в одну.
пробел	Определяет пересечение, т.е. вычисляет результат только для тех ячеек, которые входят в оба диапазона.

Функция – это заранее определенная формула, которая оперирует с одним или несколькими значениями и возвращает одно значение или несколькими. Например, можно суммировать ряд значений ячеек с помощью функции СУММ(), при этом достаточно выделить нужный диапазон. Можно записать = А1 + А2 + А3 + А4 + А5 или = СУММ(А1:А5).Т.к. функция СУММ() является часто используемой, то *Excel* имеет на панели **Редактирование** специальную кнопку для ввода этой функции Σ.

Вставка функций в формулы можно повести несколькими способами:

1. После нажатия знака равно (=) в поле имени в левой части строки формул появляется последняя используемая функция. Если раскрыть список, находящийся рядом, то можно увидеть несколько функций и надпись **Другие функции…**, через которую можно перейти к остальным.

2. Функции можно вызывать специальной кнопкой f_x на панели **Библиотека функций** вкладки **Формулы**. На этой же панели функции представлены в виде распределенных по категориям групп: Автосумма, Недавно использовались, Финансовые, Логические, Текстовые, Дата и время, Ссылки и массивы, Математические, Другие функции.

При выборе функции необходимо указать ее категорию и, используя полосу прокрутки, щелкнуть на интересующей. В открывающемся окне мастера функций дается их краткое описание в нижней части окна. После выбора функции окно мастера функций преобразуется в окно функции с полем (полями) для ввода аргументов. Поля с пояснениями, выделенными жирно, обязательны для заполнения. По окончании ввода функции необходимо нажать кнопку *Ok* или клавишу *Enter*.

Можно применять ссылки на ячейки целого диапазона листов в книге. Если все листы с данными имеют одинаковую структуру (т.е. идентичные данные в одной и той же ячейке), то для суммирования этих данных на итоговом листе можно использовать формулу с объемной ссылкой, например, = СУММ(Лист1:Лист10!А1). Результат формулы, которую *Excel* не может вычислить, будет представлен ошибочным значением (табл. 5).

Таблица 5

Ошибочные значения в формулах

Ошибочное значение	Описание
#Дел/0!	Попытка деления на ноль. Обычно связана с тем, что создается формула, в которой делитель ссылается на пустую ячейку.
#Имя?	В формуле используется имя, отсутствующее в списке имен окна диалога **Присвоение имени.** Или строка символов не заключена в двойные кавычки.
#Знач!	Введена математическая формула, которая ссылается на текстовое значение.
#Ссылка!	Отсутствует диапазон значений, на который ссылается формула.
#Н/Д	Нет данных для вычислений.
#число	Задан неправильный аргумент функции, или значение формулы слишком велико или слишком мало.
#пусто!	В формуле указано пересечение диапазонов, но они не имеют общих точек.

Массивы можно использовать для создания формул, которые возвращают некоторое множество результатов или оперируют множеством значений.

Диапазон массива – это блок ячеек, который имеет общую формулу мас- сива, т.е. туда помещается результат вычислений.

Формула массива использует несколько множеств значений, называемых **массивами аргументов,** и возвращает одно или несколько значений.

Массив констант – это специальным образом организованный список констант значений, которые можно использовать в качестве аргумента в формулах массива, т.е. формула массива использует массивы аргумента.

Например, разместим в ячейках A1:C3 массив констант, т.е. исходные данные, а в ячейках A5:C7 – диапазон массива – запишем формулу массива и получим результат вычисления (рис. 19).

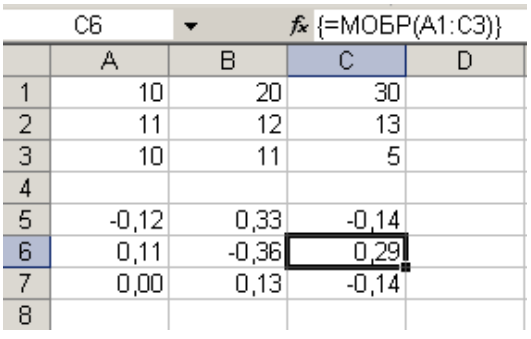


Рис. 19. Пример заполнения формулы массива

Для ввода формулы во все ячейки надо:

1. Указать диапазон массива, т.е. выделить ячейку или диапазон ячеек, который будет содержать результат.

2. Набрать знак равенства, ввести функции и адреса ячеек.

3. Если нужно указать диапазон констант массива, выделить диапазон с

помощью мыши.

4. Нажать *Ctrl* + *Shift* + *Enter* для фиксации ввода формулы массива. *Excel* заключает формулу в фигурные скобки в строке формул, указывая, что она является формулой массива. В ручную скобки не вводятся.

Для создания диаграммы необходимо воспользоваться инструментами панели Диаграммы вкладки Вставка, где можно выбрать нужный тип диаграммы. Линейчатые диаграммы обычно используют для сравнения по некоторым статьям или признакам, а гистограммы – для наблюдения изменений во времени. Круговые диаграммы наглядно отображают соотношение частей и целого. Диаграммы с областями и графики позволяют наилучшим образом изобразить непрерывное изменение величин во времени. Если не устраивает ни один из предложенных вариантов диаграмм, то необходимо воспользоваться кнопкой вызова окна панели **Диаграммы**.

После этого надо указать диапазон данных для построения диаграммы. Если данные берутся из всей таблицы, то достаточно указать любую ячейку таблицы. Если надо выбрать лишь определенные данные из таблицы, то надо выделить этот диапазон. Во время выделения можно пользоваться кнопками *Shift, Ctrl*. Для взаимной замены данных на осях надо воспользоваться кнопкой **Строка/Столбец**. После вставки диаграммы в окне *Excel* появляется контекстный инструмент **Работа с диаграммами**, содержащий три вкладки **Конструктор**, **Макет**, **Формат**.

Во вкладке **Макет** в разделе **Подписи** нужно щелкнуть на кнопке **Название диаграммы**, задать название и выбрать расположение. В меню **Названия осей** можно задать комментарии к горизонтальной и вертикальной осям, а так же их расположение. На панели **Оси** можно выбрать расположение линий сетки.

Диаграмму размещают на отдельном (с именем **Диаграмма1**) или имеющемся листе (на листе, где расположены данные для построения). Построенный график можно редактировать, растягивая, сжимая, помещая на отдельные окна; изменяя тип, размер, цвет шрифтов; тип, цвет толщину линии графика и другие параметры. При этом выгодно использовать контекстное меню и все возможности панели инструментов.

Мощным средством анализа данных *Excel* является надстройка **Поиск решения** (*Excel Solver*) (рис. 20). Она является частью блока задач, который иногда называют анализ "что-если". Процедура поиска решения позволяет найти оптимальное значение (минимальное, максимальное или равное какой-либо величине) формулы, содержащейся в ячейке, которая называется целевой. Эта процедура работает с группой ячеек, прямо или косвенно связанных с формулой в целевой ячейке. Чтобы получить по формуле, содержащейся в целевой

ячейке, заданный результат, процедура изменяет значения во влияющих ячейках. Чтобы сузить множество значений, используемых в модели, применяются ограничения, причем не обязательно, чтобы при этом использовались те же влияющие ячейки. Для расчета заданного значения применяются различные математические методы поиска. Вы можете установить режим, в котором полученные значения переменных автоматически заносятся в таблицу. Кроме того, результаты работы программы могут быть оформлены в виде отчета.

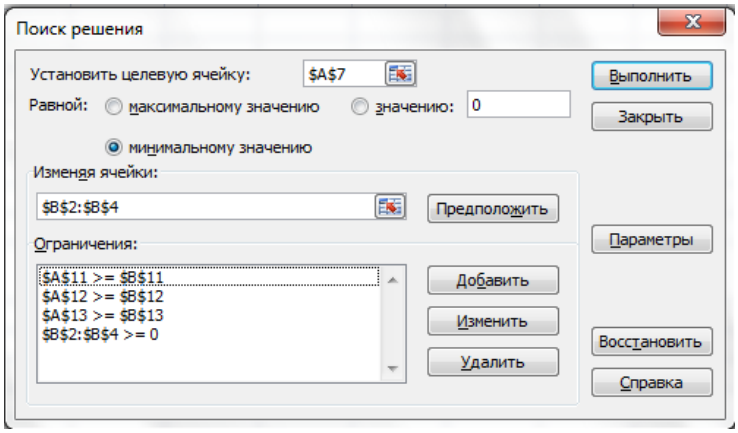


Рис. 20. Окно надстройки Поиск решения

Процедуру поиска решения можно использовать для определения значения влияющей ячейки, которое соответствует экстремуму зависимой ячейки – например, можно изменить объем планируемого бюджета рекламы и увидеть, как это повлияет на проектируемую сумму расходов.

Размер задачи, которую можно решить с помощью базовой версии этой программы, ограничивается такими предельными показателями:

- количество неизвестных – 200;
- количество формульных ограничений на неизвестные – 100;
- количество предельных условий на неизвестные – 400.

По умолчанию в *Excel* надстройка **Поиск решения** отключена. Чтобы активизировать ее, нужно щелкнуть значок **Кнопка Microsoft Office**, **Параметры Excel**, а затем выбрать категорию **Надстройки**. В поле **Управление** выбрать значение **Надстройки Excel** и нажать кнопку **Перейти**. В поле **Доступные надстройки** установить флажок рядом с пунктом **Поиск решения** и нажать кнопку *Ok*.

Для работы с процедурой **Поиск решения** необходимо выполнить следующие условия:

1. Создать таблицу с формулами, которые устанавливают связи между ячейками.

2. Выделить целевую ячейку, которая должна принять необходимое значение, и выбрать команду **Поиск решения** панели **Анализ** вкладки **Данные**.

3. Установить переключатели **Равной**, задающие значение целевой ячейки, напозицию **максимальному значению**, **минимальному значению** или **значению.** В последнем случае ввести числовое значение в поле справа.

4. Указать в поле **Изменяя ячейки**, в каких ячейках программа должна изменять значения в поисках оптимального результата.

5. Создать ограничения в списке **Ограничения**, для чего этого щелкнуть на кнопке **Добавить** и в диалоговом окне **Добавление ограничения** определить ограничение.

6. Щелкнуть на кнопке **Параметры**, и в появившемся окне установить переключатель **Неотрицательные значения** (если переменные должны быть позитивными числами), **Линейная модель** (если задача, относится к линейным моделям).

7. Щелкнув на кнопке **Выполнить**, запустить процесс поиска решения.

8. Когда появится диалоговое окно **Результаты поиска решения**, выбрать переключатель **Сохранить найденное решение** или **Восстановить исходные значения** и щелкнуть на кнопке *Ok.*

Список – совокупность строк листа, содержащих однотипные данные, например, имена клиентов и их телефонные номера или реквизиты поступивших счетов. Список может использоваться в качестве базы данных, в которой записи соответствуют строкам списка, поля – столбцам. Данные в столбцах списка должны быть одного типа, например, текст в одном столбце и числа – в следующем. Работать со списком легче, если он имеет заголовки столбцов.

Фильтр списка – способ отображения только тех строк, которые удовлетворяют заданным условиям отбора. Фильтры бывают двух типов: обычный фильтр (его еще называют автофильтр) и расширенный фильтр. В *Microsoft Excel* для фильтрации используется команда **Фильтр** на панели **Сортировка и фильтр** во вкладке **Данные**.

Чтобы отфильтровать список с помощью **Фильтра**, столбцы списка должны иметь заголовки. Перед применением **Фильтра** сначала необходимо указать ячейку в фильтруемом списке. В фильтре условия отбора задаются с помощью кнопок со стрелками, расположенными в заголовках столбцов списка. Поля, по которым установлен фильтр, отображаются со значком воронки. Если подвести указатель мыши к такой воронке, то будет показано условие фильтрации.

Применяют команду **Фильтр**, чтобы быстро отфильтровать данные с одним или двумя условиями, накладываемыми на ячейки отдельного столбца. При фильтрации списка по одному из значений, встречающихся в столбце, выбирают требуемые значения из развернувшегося набора. Для формирования более сложных условий отбора предназначены пункты **Текстовые фильтры** или **Числовые фильтры**. В окне **Пользовательский автофильтр** необходимо на-

строить окончательные условия фильтрации.

При большом объеме списка нужно снять в его верхней части флажок **Выделить все** и выбрать конкретные числа, по которым требуется выполнить отбор.

Для фильтрации списка с наложением двух условий отбора значений ячеек отдельного столбца применяется переключатель **И или ИЛИ.** При этомвыбираются из разворачивающихся наборов в полях второго условия необходимые оператор и значение сравнения.

Чтобы очистить фильтр для столбца, необходимо нажать кнопку фильтра в заголовке столбца, а затем выбрать команду **Снять фильтр** со столбца и указать **Имя столбца**.

Расширенный фильтр удобно использовать в случаях, когда результат отбора желательно поместить отдельно от основного списка, и критерии отбора задаются на рабочем листе. Для этого надо выполнить следующие действия:

1. Скопировать и вставить на свободное место шапку списка.

2. В соответствующем поле (полях) задать критерии фильтрации.

3. Установить на основной список фильтр, используя кнопку **Фильтр** на панели **Сортировка и фильтр** вкладки **Данные**.

4. На той же панели выбрать кнопку **Дополнительно**, и в появившемся окне **Расширенный фильт**р задать необходимые диапазоны ячеек.

В результате отфильтрованные данные появятся в новом списке.

Сортировка – стандартный способ упорядочивания данных одного типа. Данные можно сортировать по возрастанию или убыванию в алфавитном порядке, по датам или по величине чисел.

Строки в списке можно сортировать по значениям ячеек одного или нескольких столбцов. Кроме этого, список можно сортировать по строкам. Строки, столбцы или отдельные ячейки в процессе сортировки переупорядочиваются в соответствии с заданным пользователем порядком сортировки.

Для упорядочения ячеек по значениям (без учета формата) предусмотрен определенный порядок сортировки. При сортировке по возрастанию в *Microsoft Excel* используется следующий порядок:

1. Числа сортируются от наименьшего отрицательного до наибольшего положительного числа.

2. Текст, в том числе содержащий числа, сортируется в следующем порядке: 0 1 2 3 4 5 6 7 8 9 ' - (пробел) ! " # $ % & ( ) * , . / : ; ? @ [\ ] ^ _ { | } ~ + < = > A B C D E F G H I J K L M N O P Q R S T U V W X Y Z А Б В Г Д Е Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я.

3. Логическое значение ЛОЖЬ предшествует значению ИСТИНА.

4. Все ошибочные значения равны.

5. Пустые ячейки всегда помещаются в конец списка.

При сортировке по убыванию все перечисленные порядки заменяются на обратные за исключением пустых ячеек, которые всегда помещаются в конец списка.

В режиме простой сортировки необходимо выделить список, нажать кнопку **Сортировка и фильтр** на панели **Редактирование** вкладки **Главная** и выбрать **Сортировка от А до Я**. В этом случае список будет отсортирован по первому столбцу.

Если надо отсортировать список по нескольким полям, то для этого предназначен пункт **Настраиваемая сортировка...** в этом же перечне, что и простая. Сложная сортировка подразумевает упорядочение данных по нескольким полям. Добавлять поля можно при помощи кнопки **Добавить уровень** (рис. 21). В итоге список будет отсортирован, согласно установленным параметрам сложной сортировки. Если сортировка ведется по нескольким столбцам, то строки с одинаковыми значениями в первом столбце сортируются в порядке, определяемом вторым столбцом. Строки с одинаковыми значениями в первых двух столбцах сортируются по третьему столбцу.

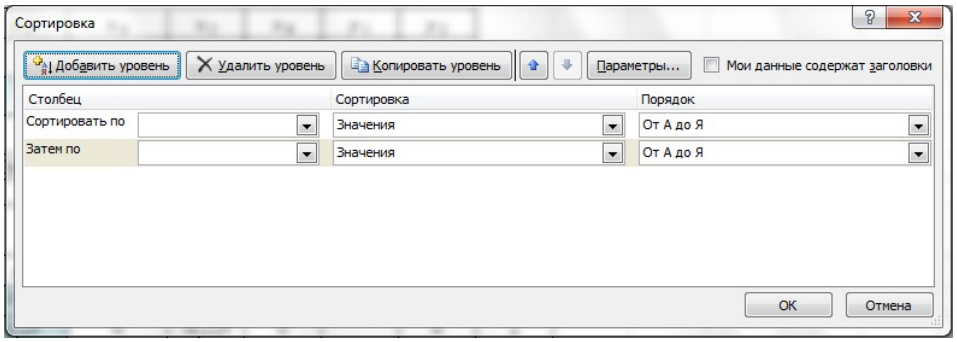


Рис. 21. Окно инструмента Сортировка

В *Microsoft Excel* предусмотрена возможность вместо сортировки по умолчанию использовать любой порядок сортировки, определенный пользователем. Если надо отсортировать поле нестандартным способом, то для этого предназначен пункт меню **Настраиваемый список..** раскрывающегося списка **Порядок**, где выбирается заранее созданный пользовательский порядок или используется стандартный.

Создание пользовательского порядка сортировки нужно перейти к **Параметрам Excel**, используя **Кнопку Microsoft Office**, а затем выбрать категорию **Основные**. В случае **Создавать списки для сортировки и заполнения** нажать кнопку **Изменить списки…** В открывшемся окне **Списки** выбрать строку **НОВЫЙ СПИСОК**. В поле **Элементы списка** набрать первый элемент создаваемого порядка сортировки, затем нажать клавишу *Enter*. Остальные элементы нужно ввести в том порядке, который необходимо использовать при сортировке. По окончании ввода нажать кнопку **Добавить**.

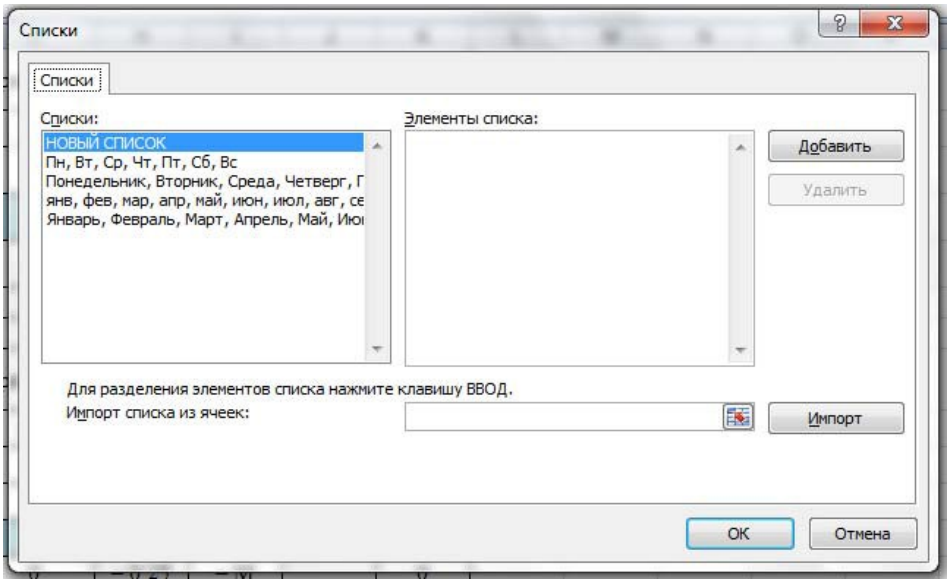


Рис. 22. Окно для создания пользовательского списка

Перемещать уровни сортировки можно при помощи кнопок **Вверх** и **Вниз**.

Не следует забывать и о контекстном меню. Из него также можно настроить сортировку и фильтрацию списка.

Вопрос 3. Презентационная графика в Microsoft PowerPoint

Программа *MS PowerPoint* является специализированным средством автоматизации для создания и оформления презентаций, призванных наглядно представить работы исполнителя группе других людей. Программа обеспечивает разработку электронных документов особого рода, отличающихся комплексным мультимедийным содержанием и особыми возможностями воспроизведения.

Любой документ *MS PowerPoint* представляет собой набор отдельных, но взаимосвязанных кадров, называемых слайдами. Каждый слайд в документе имеет собственный уникальный номер, присваиваемый по умолчанию в зависимости от места слайда. Последовательность слайдов в документе линейная. Слайды могут содержать объекты самого разного типа, например: фон, текст, таблицы, графические изображения и т.д. При этом на каждом слайде присутствует как минимум один объект – фон, который является обязательным элементом любого слайда.

Для того, чтобы узнать какое количество слайдов должна будет содержать презентация, необходимо создать план презентации, а затем разделить материал на отдельные слайды. Вероятно, понадобятся, по крайней мере, следующие слайды:

1. Основной титульный слайд.
2. Вводный слайд, содержащий основные темы или области презентации.

3. Один слайд для каждой темы или области, перечисленной на вводном слайде.

4. Итоговый слайд, повторяющий список основных тем или областей презентации.

При запуске программа *PowerPoint* открывается в режиме, называемом обычным режимом, который позволяет создавать слайды и работать с ними (рис. 23).

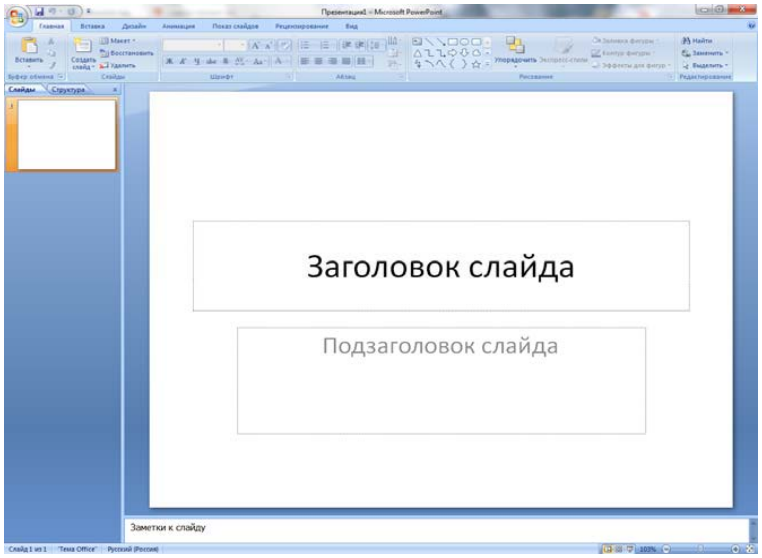


Рис. 23. Диалоговое окно приложения *MS PowerPoint*

В области **Слайд** можно работать непосредственно с отдельными слайдами. Пунктирные линии показывают места заполнения, в которые можно ввести текст или вставить изображения, диаграммы и другие объекты. Вкладка **Слайды** содержит эскизы всех полноразмерных слайдов, отображаемых в области**Слайд**. После добавления других слайдов для появления нужного слайда в области **Слайд** можно щелкнуть соответствующий эскиз на вкладке **Слайды.** Можно также перетаскивать эскизы, чтобы изменить порядок слайдов в презен- тации. Кроме того, вкладка **Слайды** позволяет добавлять и удалять слайды. Область **Заметки** позволяет ввести заметки о текущем слайде. Можно раздать заметки аудитории или обращаться к ним во время показа презентации в режиме докладчика.

По умолчанию *PowerPoint* использует для новых презентаций шаблон новой презентации, представленный на предыдущей иллюстрации. Для создания новой презентации на основе этого шаблона через**КнопкуMicrosoft Office** надо выбрать команду **Создать,** в группе **Шаблоны – Пустые и последние** и дважды щелкнуть **Новая презентация** в группе **Пустые и последние.**

После открытия шаблона новой презентации отображается только небольшая часть области **Заметки.** Чтобы увеличить видимую часть области **Заметки** и получить больше места для ввода данных, необходимо навести указатель на

верхнюю границу области **Заметки**, и когда указатель превратится в ⇳, перетащить границу вверх, чтобы увеличить область заметок докладчика. Слайд, который автоматически появляется в презентации, содержит два место заполнителя, один из которых отформатирован для заголовка, а второй – для подзаголовка.

Чтобы одновременно с добавлением слайда в презентацию выбрать макет нового слайда, можно на вкладке **Слайды** указать непосредственно под единственным содержащимся на этой вкладке слайдом. На панели **Слайды** вкладки **Главная** щелкнуть стрелку рядом с кнопкой **Создать слайд.** Появится коллекция, в которой отображаются эскизы различных доступных макетов слайдов. Имя определяет содержимое, для которого спроектирован каждый из макетов. Для нового слайда выбрать нужный макет. Новый слайд появляется и на вкладке **Слайды**, где он выделяется как текущий, и в области **Слайд**. Эту процедуру нужно повторить для каждого добавляемого слайда.

Если нужно создать два слайда, аналогичных по содержанию и макету, можно создать один слайд с форматированием и содержанием, общими для обоих слайдов, а затем скопировать этот слайд и добавить на каждый слайд окончательные индивидуальные детали.

Копирование, вставка и удаление слайдов производиться схожим образом с выполнением этих операций для текста вкладка **Главная** панель **Буфер обмена**, только в этом случае происходит выделение слайдов.

При изменении порядка расположения слайдов на вкладке **Слайды** щелкнуть слайд, который нужно переместить, а затем перетащить его в новое место.

Чтобы выделить несколько слайдов, также используются клавиши *Ctrl* и *Shift.*

Самым общим содержанием слайдов в презентации *PowerPoint* является текст – в заголовках, названиях и маркированных списках. Чтобы добавить текст на любой слайд, надо щелкнуть местозаполнитель, в который нужно добавить текст, а затем ввести или вставить нужный текст.

В одних местозаполнителях текст автоматически форматируется как маркированный список, а в других местозаполнителях – нет. На панели **Абзац** вкладки **Главная**, чтобы переключиться между маркированным и немаркированным текстом, нужно выделить текст и нажать кнопку **Маркеры**.

Существует множество способов изменить внешний вид текста на слайде, от основных кнопок вкладки **Главная**, предназначенных для форматирования характеристик шрифта, стиля, размера, цвета и абзаца, до дополнительных параметров, таких как анимация или преобразование в рисунки *SmartArt*.

Слишком большое количество текста делает слайд запутанным и непонятным для аудитории. Однако, если убрать с экрана часть данных, сделав их невидимыми для аудитории, как можно их отследить?

Решением этой проблемы являются заметки докладчика, которые можно ввести в области **Заметки** для каждого слайда. Заметки докладчика помогают в процессе презентации избавить экран от избыточного содержания, одновременно позволяя отслеживать все данные, нужные во время презентации.

Можно также в любой момент легко вырезать излишне подробный текст из области **Слайд**, а затем вставить этот текст прямо в область **Заметки**, чтобы можно было пользоваться им для справки.

PowerPoint предоставляет множество тем, упрощая изменение общего вида презентации. **Тема** представляет собой набор элементов оформления, придающий особый, единообразный внешний вид всем документам *Office*, используя конкретные сочетания цветов, шрифтов и эффектов. *PowerPoint* автоматически применяет к презентациям, созданным с помощью шаблона новой презентации, тему *Office*, но внешний вид презентации можно легко изменить в любой момент, применив другую тему. На панели **Темы** вкладки **Дизайн** можно выбрать нужную тему документа. Для предварительного просмотра внешнего вида текущего слайда после применения конкретной темы необходимо навести указатель на эскиз этой темы.

Если не указано иное, *PowerPoint* применяет темы ко всей презентации. Чтобы изменить внешний вид только выбранных слайдов, на вкладке **Слайды** нужно указать слайды, которые нужно изменить, а затем щелкнуть правой кнопкой мыши тему, которую нужно применить, и выбрать в контекстном меню команду **Применить к выделенным слайдам.**

PowerPoint позволяет добавлять множество видов аудио и видеоданных, включая таблицы, рисунки *SmartArt*, клип, фигуры, диаграммы, музыку, фильмы, звуки и анимации. Можно также добавить гиперссылки, чтобы повысить гибкость перемещения по презентации и вне ее, а также привлекающие глаз переходы между слайдами.

Для добавления клипа нужно указать прототип, в который необходимо добавить клип. Если прототип не выделен или если выделен прототип, в который нельзя вставить изображение, клип вставляется в центр слайда. Затем во вкладке **Вставка** на панели **Иллюстрации** нажать кнопку **Клип.** В области задач **Клип** необходимо выбрать нужный клип. Теперь клип можно переместить, изменить размер, повернуть, добавить текст и выполнить иные изменения.

Рисунок *SmartArt* – это визуальное представление сведений, которое можно полностью настроить. Преобразование текста в рисунок *SmartArt* – это быстрый способ преобразовать существующие слайды в профессионально оформленные иллюстрации.

Чтобы преобразовать существующий текст в рисунок *SmartArt*, нужно щелкнуть местозаполнитель, содержащий текст, который нужно преобразовать.

На панели **Абзац** вкладки **Главная** нажать кнопку **Преобразовать в рисунок *SmartArt* .**

Чтобы увидеть, как будет выглядеть рисунок *SmartArt* с нужным текстом, в коллекции указать на эскиз этого рисунка *SmartArt*. Для просмотра полного набора макетов надо выбрать кнопку **Дополнительные рисунки *SmartArt*.**

Хотя рисунок *SmartArt* легче всего создать для существующего текста, можно сначала вставить нужный рисунок *SmartArt,* а затем добавить к нему текст.

Смены слайдов представляют собой анимационные эффекты, возникающие при переходе от одного слайда к следующему. *PowerPoint* предоставляет множество типов смены слайдов, включая стандартные эффекты затухания, растворения, обрезания и стирания, а также более необычные переходы, например колеса и шахматные доски.

На панели **Переходк этому слайду** вкладки **Анимации** можно выбрать нужный вариант перехода. Для предварительного просмотра внешнего вида текущего слайда с использованием конкретного варианта перехода указать на эскиз этого перехода. Можно выбрать другие варианты в группе **Переход к этому слайду,** чтобы управлять скоростью перехода, добавить звук и применить этот же вариант перехода ко всем слайдам презентации.

Для перехода с одного слайда на другой, к ресурсу в локальной сети или в Интернете либо даже к другому файлу или программе можно воспользоваться гиперссылками.

Выделить текст, который нужно щелкнуть для активации гиперссылки либо можно выделить объект (например, клип или рисунок *SmartArt*), затем на панели **Ссылки** вкладки **Вставка** выбрать элемент **Гиперссылка**. В диалоговом окне **Вставка гиперссылки** надо нажать соответствующую кнопку в поле **Мои адреса,** чтобы задать назначение ссылки (то есть место, на которое указывает ссылка).

Для просмотра презентации на экране компьютера в том виде, в каком она будет представлена аудитории, необходимо на панели **Начать показ слайдов** вкладки **Показ слайдов** выполнить одно из следующих действий:

- для запуска презентации с первого слайда выбрать **С начала** или нажать клавишу *F*5;
- чтобы начать показ со слайда, в настоящий момент находящегося в области Слайд, выбрать **С текущего слайда**;

Презентация открывается в режиме показа слайдов. Для перехода к следующему слайду нужно щелкать мышью или нажимать клавишу *Enter*.Чтобы вернуться в обычный режим, в любой момент можно нажать клавишу *Esc*.

Вопрос 4. Файлы и файловая система

Общее управление компьютером осуществляется на основе командного языка (языка директив), с помощью которого можно осуществлять такие операции, как разметка дисков, копирование файлов, распечатка каталогов на экране дисплея и другие действия.

Операционная система содержит следующие основные компоненты:

- файловую систему;
- интерпретатор командного языка (командный процессор);
- драйверы внешних устройств.

Файловая система – это совокупность файлов данных и программ, размещенных на технических носителях в соответствии с определенным набором правил. Структура файловой системы определяет удобство работы, скорость доступа к файлам и т.д.

Файл – это именованная совокупность элементов информации, хранящаяся на электронных носителях. Имена файлов записываются следующим образом: ИМЯ.ТИП, где ИМЯ набор символов латинского или русского алфавита, цифр и специальных символов ~' & @ ( ) % _ # ` $, а ТИП или РАСШИРЕНИЕ файла состоит не более чем из 3-4 символов. В отличие от имени тип может отсутствовать в спецификации файла.

Тип файла используется для классификации, определения принадлежности к какой-то группе с общими свойствами. Например, тип *DOC, TXT* – текстовые файлы, *EXE, COM* – файлы, содержащие программы, готовые к выполнению, *PAS, BAS, ASM* – программы, написанные на алгоритмических языках Паскаль, Бейсик и Ассемблер. Для ОС безразлично, какими строчными или заглавными буквами записывается файл.

Имя и тип файла не обеспечивают всех потребностей, которые возникают при работе с файлами. Чтобы указать отличительные особенности некоторых файлов, вводится понятие атрибута файла. Имеются следующие атрибуты: архивный (*A*); только для чтения (*R*); системный (*S*); скрытый (*H*). Атрибут архивный (*A*) присваивается файлу для того, чтобы его можно было проще разыскать в подкаталогах при создании копий, обновлении старых файлов и реализации подобных операций, выполняемых с помощью некоторых команд. Эти файлы по внутренней структуре ничем не отличаются от обычных, кроме атрибута *A*. Их не следует путать с архивированными файлами, получаемыми с помощью специальных программ-архиваторов. Файлы «только для чтения» защищены от изменения и случайного стирания. Системные файлы обеспечивают работу операционной системы. Имена скрытых файлов и их характеристики при просмотре каталогов не выводятся на экран дисплея.

При использовании имен файлов в качестве параметров команд необходимо указывать адрес или путь к файлу. **Путем к файлу** называется цепочка символов, начиная с имени дисковода, корневого каталога и последующих подкаталогов вплоть до каталога, содержащего необходимый файл.

Имя дисковода – это одна из букв латинского алфавита. Персональный компьютер имеет несколько накопителей на магнитных носителях, исходя из этого принято обозначать: *A* – гибкие диски, *C*: *D*: и т.д. – жесткие диски, самыми последними указываются дисководы типа *CD-* и *DVD-ROM*.

Каталог – это справочник файлов и библиотек со ссылками на их расположение, содержащее информацию о файлах (имя, тип, размеры в байтах, дата и время создания, атрибуты) и других каталогах, называемых подкаталогами, используется операционной системой для определения местоположения файла.

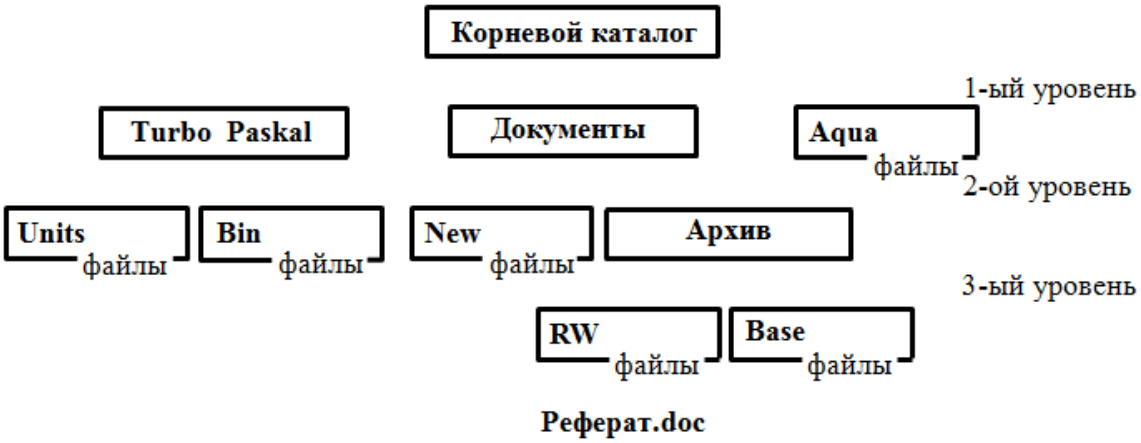


Рис. 24. Пример иерархического строения диска

На каждом диске имеется один главный или **корневой каталог**. Каталоги, входящие в корневой каталог называются подкаталогами 1-го уровня. Каталоги, входящие в состав подкаталога 1-го уровня, называются подкаталогами 2-го уровня и т.д. Каждый подкаталог является оглавлением, содержащим перечень имен файлов и подкаталогов, возможны варианты, когда в оглавлении имеются только имена файлов. Организация файловой системы в виде подкаталогов удобна для сортировки информации по различным темам. Разбиение на подкаталоги зависит от желания пользователя компьютером. Иерархическое строение диска можно представить в виде дерева подкаталогов.

Все имена подкаталогов отделяются друг от друга символом \. Предположим, что корневой каталог находится на диске *D* и, используя принятые обозначения, обратимся, для примера, к файлу Реферат.doc – *D*:\Документы\Архив*RW*\Реферат.doc.

В случае поиска файла при задании его имени можно использовать символы * и ?. Применение звездочки * для описания имени указывает на то, что имя

может состоять из любых символов, безразличных для команд операционной системы. Например: *.txt для операционной системы означает, что будут обрабатываться файлы с любым именем, которые имеют тип .txt. Запись *.* определяет все файлы, находящиеся на магнитном носителе. Знак вопроса ? в имени файла означает, что на его месте может находиться произвольный символ.

ТЕМА 5. ТЕОРИЯ БАЗ ДАННЫХ

План:

1. Общие понятия о базах данных и системах управления ими.
2. Классификация моделей данных.
3. Журнализация.
4. Создание баз данных в Microsoft Access.
   4.1 Знакомство с Access. Создание таблиц.
   4.2 Создание связей между таблицами.
   4.3 Отбор данных с помощью запросов.
   4.4 Использование форм в базе данных.
   4.5 Создание отчетов.

Вопрос 1. Общие понятия о базах данных и системах управления базами данных

Всегда, когда возникает потребность манипулировать большими массивами данных, используются базы данных. **База данных** (БД) – это, прежде всего, набор таблиц, хотя в базу данных могут входить также процедуры и ряд других объектов.

С точки зрения пользователя, база данных – это программа, которая обеспечивает работу с информацией. При запуске такой программы на экране, как правило, появляется таблица, просматривая которую пользователь может найти интересующие его сведения. Если система позволяет, то он может внести изменения в базу данных: добавить новую информацию или удалить ненужную.

С точки зрения программиста, база данных – это набор файлов, содержащих информацию. Разрабатывая базу данных для пользователя, программист создает программу, которая обеспечивает работу с файлами данных.

В настоящее время существует достаточно большое количество программных систем, позволяющих создавать и использовать локальные (*dBASE*, *FoxPro*, *Access*, *Paradox*) и удаленные (*Interbase*, *Oracle*, *Sysbase*, *Infomix*, *MicrosoftSQLServer*) базы данных.

Банк данных является современной формой организации хранения и доступа к информации. **Банк данных** – это, прежде всего, система специальным образом организованных данных (баз данных), программных, технических, языковых, организационно-методических средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.

Банк данных является сложной системой, включающей в себя все обеспечивающие подсистемы, необходимые для функционирования любой системы

автоматизированной обработки. Базы данных создаются не для решения какой-либо одной задачи для одного пользователя, а для многоцелевого использования.

Другая отличительная особенность банков данных – наличие специальных языковых и программных средств, облегчающих для пользователей выполнение всех операций, связанных с организацией хранения данных, их корректировки и доступа к ним. Такая совокупность языковых и программных средств называется системой управления базой данных (СУБД).

Использование банков данных при правильной его организации должно существенно изменить деятельность организации, где он внедряется, привести к сокращению документооборота, форм документов, перераспределению функций между сотрудниками.

Централизованное управление данными также дает целый ряд преимуществ. Использование банков данных приводит к сокращению трудоемкости создания системы и снижению требований к остальным участникам функционирования банков данных.

Банк данных обеспечивают возможность более полной реализации принципа независимости прикладных программ от данных, чем это возможно при организации локальных файлов.

Наличие в составе СУБД средств, ориентированных на разные категории пользователей, делает возможной работу с базой данных не только профессионалов в области в области обработки данных, но и практически любого, причем это использование может быть как для их профессиональных целей, так и для удовлетворения потребности в информации в быту и т.п.

Банк данных является сложной человеко-машинной системой, включающей в свой состав различные взаимосвязанные и взаимозависимые компоненты. Ядром банка данных является база данных. К банкам данных не относятся немашинные документы, служащие источником информации, вводимой в БД, файлы входной и выходной информации, архивные файлы, выходные документы. Однако многие СУБД включают в свой состав языковые средства для описания этих компонентов.

Технические средства банков данных представляют универсальные ЭВМ, периферийные средства для ввода информации в базу данных и отображения выводимой информации. Если банк данных реализуется в сети, то необходимы соответствующие технические средства для обеспечения ее работы. Программные средства представляют собой сложный комплекс, обеспечивающий взаимодействие всех частей информационной системы при ее функционировании. Основу программных средств банков данных составляют СУБД.

Важной компонентой СУБД являются трансляторы или компиляторы для

используемых ею языковых средств. Языковые средства банков данных являются важнейшей компонентой, так как в конечном итоге они обеспечивают интерфейс пользователей разных категорий с банком данных.

Организационно-методические средства банка данных представляют собой инструкции, методические и регламентирующие материалы, предназначенные для пользователей разных категорий, взаимодействующих с банком данных.

Администраторы банка данных – это группа специалистов, обеспечивающих создание, функционирование и развитие банка данных. Такая группа специалистов считается составной часть банка данных.

СУБД позволяют структурировать, систематизировать и организовывать данные для их компьютерного хранения и обработки. Именно системы управления базами данных являются основой практически любой информационной системы.

СУБД можно определить как некую систему управления данными, обладающую следующими свойствами:

- поддержание логически согласованного набора файлов;
- обеспечение языка манипулирования данными;
- восстановление информации после разного рода сбоев;
- обеспечение параллельной работы нескольких пользователей.

К основным функциям, выполняемым системами управления базами данных, обычно относят следующие:

- непосредственное управление данными во внешней памяти;
- управление буферами оперативной памяти;
- управление транзакциями;
- протоколирование;
- поддержка языков баз данных.

Функция непосредственного управления данными во внешней памяти включает обеспечение необходимых структур внешней памяти как для хранения данных, непосредственно входящих в базу данных, так и для служебных целей, например для ускорения доступа к данным в некоторых случаях. В зависимости от способа реализации СУБД может либо использовать возможности существующих файловых систем, либо работать с устройствами внешней памяти на низком уровне.

Объем информации, хранящейся в базе данных, с которой работает СУБД, обычно достаточно велик и практически всегда превышает доступный объем оперативной памяти. При этом время доступа к данным, хранящимся в оперативной памяти, существенно меньше, чем к данным, хранящимся на устройствах внешней памяти. Увеличения скорости обмена данными можно достичь, используя буферизацию данных в оперативной памяти. В СУБД обычно под-

держивается собственный набор буферов оперативной памяти с собственным механизмом замены буферов.

Транзакцией называется последовательность операций над базой данных, рассматриваемых СУБД как единое целое. Если все операции успешно выполнены, то транзакция также считается успешно выполненной и СУБД фиксирует все изменения данных, произведенные этой транзакцией (то есть заносит изменения во внешнюю память). Если же хотя бы одна операция транзакции заканчивается неудачей, то транзакция считается невыполненной и производится отмена всех изменений данных, произведенных в ходе выполнения транзакции, и возврат базы данных к состоянию до начала выполнения транзакции. Управление транзакциями необходимо для поддержания логической целостности базы данных. При соответствующем управлении параллельно выполняющимися транзакциями со стороны СУБД каждый из пользователей может не ощущать задержки при выполнении своих команд.

Вопрос 2. Классификация моделей данных

Ядром любой БД является модель данных. **Модель данных** – это совокупность структур данных и операций их обработки. Так как СУБД имеет 3-х уровневую архитектуру, то понятие модели данных связано с каждым уровнем.

Физическая модель данных связана с организацией внешней памяти и структур хранения, используемых в данной операционной среде. На концептуальном уровне модели данных наиболее важны для разработчиков БД, так как именно ими определяется тип СУБД. Для внешнего уровня отдельных моделей данных нет, они лишь являются подсхемами концептуальных моделей данных.

Кроме моделей данных, соответствующих трем уровням архитектуры СУБД, существуют предшествующие им, не связанные с компьютерной реализацией. Они служат переходным звеном от реального мира к БД. Это класс инфологических (семантических) моделей.

Общая классификация моделей данных:

1. Инфологические (семантические) модели используются на ранних стадиях проектирования баз данных для формального описания предметной области. Они содержат информацию о классах объектов, их свойствах и взаимосвязях, описания структур данных без привязки к какой-либо конкретной СУБД. Этот вид моделей может быть представлен:

- диаграммой Бахмана;
- моделью сущность-связь (*ER*-модель).

2. Физическая модель данных оперирует категориями, касающимися организации внешней памяти и структур хранения, используемых в данной опера-

ционной среде. В настоящий момент в качестве физических моделей используются различные методы размещения данных, основанные на файловых структурах: это организация файлов прямого и последовательного доступа, индексных файлов и инвертированных списков. Кроме того, современные СУБД широко используют страничную организацию данных. В этом случае база данных представлена минимальным количеством файлов, а задачи поиска, чтения и записи данных выполняет сама СУБД, а не операционная система.

3. Даталогические модели являются моделями концептуального уровня, разрабатываются для конкретной СУБД и, в свою очередь, делятся на:

а) документальные моделиданных соответствуют слабоструктурированной информации, ориентированной на свободные форматы документов на естественном языке:

- ориентированные на формат документа модели связаны со стандартным общим языком разметки *SGML* (*Standart Generaliset Markup Language*), а также *HTML*, предназначенным для управления процессом вывода содержимого документа на экран;

- дескрипторные модели – в них каждому документу соответствует дескриптор-описатель, который имеет жесткую структуру и опи- сывает документ в соответствии с заранее определенными характеристиками;

- тезаурусные модели данных основаны на принципе организации словарей, содержат языковые конструкции и принципы их взаимодействия в заданной грамматике;

б) фактографические модели делятся на:

- объектно-ориентированныемодели перекликаются с семантическими моделями данных, и принципы их похожи на принципы объектно-ориентированных языков программирования. Структура таких моделей графически представима в виде дерева, узлами которого яв- ляются объекты, а свойства объектов описываются типом;

- теоретико-графовые модели делятся на:

- иерархические – объекты иерархической модели данных связаны иерархическими отношениями и образуют ориентированный граф. Основные понятия иерархических структур: уровень, узел (совокупность свойств данных, описывающих объект), связь;
- сетевыемодели данных при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом;

- теоретико-множественные модели делятся на:
  - реляционные – данные представлены только в виде таблиц;
  - бинарных ассоциаций (инвертированных списков).

Чаще всего рассматриваются реляционные модели данных, так как в последнее время реляционные СУБД заняли преимущественное положение, поскольку их недостатки связаны в основном с техническими проблемами и компенсируются ростом быстродействия и ресурсов памяти современных ЭВМ.

Вопрос 3. Журнализация

Одним из основных требований к СУБД является надежность хранения данных во внешней памяти. Под надежностью хранения понимается то, что СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя. Аппаратные сбои обычно подразделяются на два вида:

- мягкие сбои связаны с внезапной остановкой работы компьютера. Обычно являются следствием внезапного выключения питания или «зависания» операционной системы (что особенно характерно для операционных систем *Windows*);
- жесткие сбои характеризуются потерей информации на носителях внешней памяти.

Программные сбои обычно возникают вследствие ошибок в программах. Причем эти ошибки могут быть как в самой СУБД, что может привести к аварийному завершению ее работы, так и в пользовательской программе. Первый случай можно рассматривать как разновидность мягкого аппаратного сбоя. Во втором случае незавершенной остается только одна транзакция.

В любом случае для восстановления информации в базе данных необходимо иметь некоторую дополнительную информацию. Таким образом, для поддержания надежности хранения данных требуется избыточность данных. Причем та часть информации, которая используется для восстановления, должна храниться особо надежно. Наиболее распространенным методом поддержания такой избыточной информации является ведение журнала изменений базы данных.

Журнал представляет собой особую часть базы данных, недоступную пользователям СУБД и поддерживаемую с особой тщательностью (иногда используются две копии журнала, располагаемые на разных дисках), в которую поступают записи обо всех изменениях основной части базы данных. В разных СУБД изменения базы данных журнализируются на разных уровнях: иногда запись в журнале соответствует некоторой логической операции изменения базы данных, иногда – минимальной внутренней операции модификации страницы внешней памяти. Могут также использоваться одновременно оба подхода.

Во всех случаях придерживаются стратегии «упреждающей» записи в журнал (протокол *Write Ahead Log – WAL*). Эта стратегия заключается в том,

что запись об изменении любого объекта базы данных должна быть занесена в журнал до того, как будет выполнено и зафиксировано изменение этого объекта. Если в СУБД корректно соблюдается протокол *WAL*, то с помощью журнала можно решить все проблемы восстановления базы данных после любого сбоя.

Самая простая ситуация восстановления – индивидуальный откат транзакции. Строго говоря, для этого не требуется общесистемный журнал изменений базы данных. Достаточно для каждой транзакции поддерживать локальный журнал операций модификации базы данных, выполненных в этой транзакции, и производить откат транзакции, путем выполнения обратных операций, следуя от конца локального журнала. Но в большинстве систем локальные журналы не поддерживают, а индивидуальный откат транзакции выполняют по общесистемному журналу, для чего все записи, относящиеся к одной транзакции, связывают обратным списком (от конца к началу).

При мягком сбое во внешней памяти основной части базы данных могут находиться объекты, модифицированные транзакциями, не закончившимися к моменту сбоя, и могут отсутствовать объекты, модифицированные транзакциями, которые к моменту сбоя успешно завершились. При соблюдении протокола *WAL* во внешней памяти журнала должны гарантированно находиться записи, относящиеся к операциям модификации обоих видов объектов. Целью процесса восстановления после мягкого сбоя является приведение внешней памяти основной части базы данных в такое состояние, которое возникло бы при фиксации во внешней памяти изменений всех завершившихся транзакций и которое не содержало бы никаких следов незаконченных транзакций. Для того чтобы этого добиться, сначала производят откат незавершенных транзакций, а потом повторно воспроизводят те операции завершенных транзакций, результаты которых не отображены во внешней памяти.

Для восстановления базы данных после жесткого сбоя используют журнал и архивную копию базы данных. Архивная копия – это полная копия базы данных к моменту начала заполнения журнала. Для нормального восстановления базы данных после жесткого сбоя, естественно, необходимо, чтобы журнал не пропал. Тогда восстановление базы данных состоит в том, что, исходя из архивной копии, по журналу воспроизводится работа всех транзакций, которые закончились к моменту сбоя.

Для работы с информацией, хранящейся в базе данных, используются специальные языки, носящее общее название языков баз данных. Чаще всего выделяются два языка:

- язык определения схем данных (*Schema Definition Language, SDL*) служит главным образом для определения логической структуры базы данных;
- язык манипулирования данными (*Data Manipulation Language, DML*) со-

держит набор операторов манипулирования данными, то есть операторов, позволяющих заносить данные в базу, а также удалять, модифицировать или выбирать существующие данные.

В современных СУБД поддерживается единый интегрированный язык, содержащий все необходимые средства для работы с базой данных, начиная от ее создания, и обеспечивающий базовый пользовательский интерфейс с базами данных. Стандартным языком наиболее распространенных в настоящее время реляционных СУБД является язык *SQL* (*Structured Query Language*). Таким образом, указанные ранее языки баз данных на сегодняшний день фактически являются подмножествами единого стандартного языка SQL.

Язык *SQL* позволяет определять схему реляционной базы данных и манипулировать данными. При этом именование объектов базы данных (для реляционной базы данных – именование таблиц и их полей) поддерживается на языковом уровне в том смысле, что компилятор языка *SQL* производит преобразование имен объектов в их внутренние идентификаторы на основании специально поддерживаемых служебных таблиц-каталогов.

Язык *SQL* содержит специальные средства определения ограничений целостности базы данных. Ограничения целостности хранятся в специальных таблицах-каталогах, и обеспечение контроля целостности базы данных производится на языковом уровне – при компиляции операторов модификации базы данных компилятор *SQL* на основании имеющихся в базе данных ограничений целостности генерирует соответствующий программный код.

Специальные операторы языка *SQL* позволяют определять так называемые представления базы данных, фактически являющиеся хранимыми в базе данных запросами (результатом любого запроса к реляционной базе данных является таблица) с именованными столбцами, называемыми полями. Для пользователя представление является такой же таблицей, как любая базовая таблица, хранимая в базе данных, но с помощью представлений можно ограничить или, наоборот, расширить видимость данных для конкретного пользователя. Поддержка представлений производится также на языковом уровне.

Наконец, авторизация доступа к объектам базы данных производится также на основе специального набора операторов *SQL*. Идея состоит в том, что для выполнения операторов *SQL* разного вида пользователь должен обладать различными полномочиями. Пользователь, создавший таблицу базы данных, обладает полным набором полномочий для работы с данной таблицей. В число этих полномочий входит полномочие на передачу всех или части полномочий другим пользователям, включая полномочие на передачу полномочий. Полномочия пользователей описываются в специальных таблицах-каталогах, контроль полномочий поддерживается на языковом уровне.

Вопрос 4. Создание баз данных в Microsoft Access

4.1 Знакомство с Access. Создание таблиц

База данных (БД) – упорядоченная совокупность данных, предназначенных для хранения, накопления и обработки с помощью ЭВМ. Для создания и ведения баз данных (их обновления, обеспечения доступа по запросам и выдачи данных по ним пользователю) используется набор языковых и программных средств, называемых *системой управления базами данных (СУБД).*

Объекты базы данных Access

К объектам базы данных Access относятся:

1. *Таблицы* – предназначены для упорядоченного хранения данных.

2. *Запросы* – предназначены для поиска, извлечения данных и выполнения вычислений.

3. *Формы* – предназначены для удобного просмотра, изменения и добавления данных в таблицах.

4. *Отчеты* – используются для анализа и печати данных.

5. *Страницы доступа к данным* – предназначены для просмотра, ввода, обновления и анализа данных через сеть или из любого места компьютера.

6. *Макросы* – используются для выполнения часто встречающегося набора макрокоманд, осуществляющих обработку данных.

7. *Модули* – предназначены для описания инструкций и процедур на языке VBA.

Основным объектом базы данных является таблица, которая состоит из записей (строк) и полей (столбцов). На пересечении записи и поля образуется ячейка, в которой содержатся данные.

Каждому полю таблицы присваивается *уникальное имя*, которое не может содержать более 64 символов. В каждом поле содержатся данные одного типа.

Таблица 6

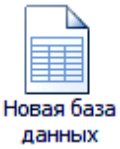
Типы данных

Тип	Описание
Текстовый	Используется для хранения символьных или числовых данных, не требующих вычислений. В свойстве *Размер поля* задается максимальное количество символов, которые могут быть введены в данное поле. По умолчанию размер устанавливается в 50 знаков. Максимальное количество символов, которые могут содержаться в текстовом поле, – 255
Поле МЕМО	Предназначено для ввода текстовой информации, по объему превышающей 255 символов; может содержать до 65 536 символов
Числовой	Предназначен для хранения числовых данных, используемых в математических расчетах. На вкладках Общие и Подстановка можно установить свойства числового поля, среди которых Размер поля,

	Формат поля, Число десятичных знаков
Дата/Время	Используется для представления даты и времени. Выбор конкретного формата даты или времени устанавливается в свойстве Формат даты
Денежный	Предназначен для хранения данных, точность представления которых колеблется от 1 до 4 знаков после запятой. Целая часть может содержать до 15 десятичных знаков
Счетчик	Предназначен для автоматической вставки уникальных последовательных (увеличивающихся на 1) или случайных чисел в качестве номера новой записи. Номер, присвоенный записи, не может быть удален или изменен. Поля с этим типом данных используются в качестве ключевых полей таблицы
Логический	Предназначен для хранения одного из двух значений, интерпретируемых как «Да / Нет», «Истина / Ложь», «Вкл. / Выкл.»
Поле объекта OLE	Содержит данные, созданные в других программах, которые используют протокол OLE. Это могут быть, например, документы Word, электронные таблицы Excel, рисунки, звуковые и видеозаписи и др. Объекты OLE связываются с базой данных Access или внедряются в нее. Сортировать, группировать и индексировать поля объектов OLE нельзя
Гиперссылка	Специальный тип, предназначенный для хранения гиперссылок
Мастер подстановок	Предназначен для автоматического определения поля. С его помощью будет создано поле со списком, из которого можно выбирать данные, содержащиеся в другой таблице или в наборе постоянных значений

Создание базы данных

1. Запустите Microsoft Access.

2. Нажмите на кнопку 

3. Задайте имя новой базы данных – «Записная книжка.accdb».

4. На вкладке ленты *Создание* в панели инструментов *Таблицы* нажмите на кнопку *Конструктор таблиц*.

5. Введите имена полей и укажите типы данных, к которым они относятся.

Имя поля	**Тип данных**
№ п/п	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Адрес	Текстовый
Индекс	Числовой
Телефон	Текстовый
Хобби	Текстовый
Эл. почта	Гиперссылка

6. Выйдите из режима *Конструктора,* предварительно сохранив таблицу под именем *«Друзья»;* ключевые поля не задавайте.

7. Откройте таблицу *«Друзья»* двойным щелчком мыши и заполните в ней

10 строк.

8. Добавьте поля *«Отчество»* и *«Дата рождения»*, для этого:

1) установите курсор на поле, перед которым нужно вставить новый столбец;

2) выполните команду: вкладка ленты *Режим таблицы* → панель инструментов *Поля и столбцы* → *Вставить*;

3) щелкнув два раза на *Поле1,* переименуйте его в *«Отчество»*, а *Поле2 – «Дата рождения»*.

9. Перейдите в режим Конструктора командой: вкладка ленты *Главная* → *Режимы* → *Конструктор.*

10. Для поля *«Дата рождения»* установите тип данных *Дата / время*; в свойствах поля выберите *Краткий формат даты*.

11. Отформатируйте таблицу следующим образом:

4) цвет сетки – темно-красный;

5) цвет фона – голубой;

6) цвет текста – темно-красный, размер – 12 пт, начертание – курсив.

12. Переименуйте поле *«Хобби»* в *«Увлечения».*

13. Удалите запись под номером 8.

14. Измените размер ячеек так, чтобы были видны все данные. Для этого достаточно два раза щелкнуть левой кнопкой мыши на границе полей.

15. Расположите поля в следующем порядке: *«№», «Фамилия», «Имя», «Отчество», «Телефон», «Дата рождения», «Увлечения», «Адрес», «Индекс», «Фото», «Эл_почта».*

16. Заполните пустые ячейки таблицы.

17. В режиме Конструктора добавьте поле *«Семейное положение»*, в котором будет содержаться фиксированный набор значений – замужем, не замужем, женат, не женат. Для создания раскрывающегося списка будем использовать *Мастер подстановок*:

7) установите тип данных *Мастер подстановок*;

8) в появившемся диалоговом окне выберите строку *«Будет введен фиксированный набор значений»* и нажмите кнопку *Далее*;

9) число столбцов – 1;

10) введите данные списка – замужем, не замужем, женат, не женат;

11) нажмите кнопку *Готово*.

18. С помощью раскрывающегося списка заполните новый столбец. Поскольку таблица получилась широкая, то при заполнении данного столбца возникают некоторые неудобства: не видно фамилии человека, для которого заполняется поле *«Семейное положение»*. Чтобы фамилия была постоянно видна при заполнении таблицы, необходимо воспользоваться командой *Закрепить*

столбцы из контекстного меню поля *«Фамилия»*.

4.2 Создание связей между таблицами

1. Запустите Microsoft Access.

2. Создадим базу данных «Фирма». Сотрудники данной организации работают с клиентами и выполняют их заказы.

Если все сведения поместить в одной таблице, то она станет очень неудобной для работы. В ней начнутся повторы данных. Всякий раз, когда сотрудник Иванов будет работать с какой-либо фирмой, придется прописывать данные о сотруднике и клиенте заново, в результате чего можно допустить множество ошибок. Чтобы уменьшить число ошибок, можно исходную таблицу разбить на несколько таблиц и установить связи между ними. Это будет более рационально, чем прежде.

Таким образом, необходимо создать 3 таблицы: *Сотрудники*, *Клиенты* и *Заказы*.

Сотрудники

Имя поля	Тип данных
Код сотрудника	Счетчик
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Должность	Текстовый
Телефон	Текстовый
Адрес	Текстовый
Дата рождения	Дата/Время
Заработная плата	Денежный
Фото	Объект OLE
Эл_почта	Гиперссылка

Клиенты

Имя поля	Тип данных
Код клиента	Счетчик
Название компании	Текстовый
Адрес	Текстовый
Номер телефона	Текстовый
Факс	Числовой
Адрес электронной почты	Гиперссылка
Заметки	Поле МЕМО

Заказы

Имя поля	Тип данных
Код заказа	Счетчик
Код клиента	Числовой
Код сотрудника	Числовой
Дата размещения	Дата/Время
Дата исполнения	Дата/Время
Сумма	Денежный
Отметка о выполнении	Логический

3. Отдельные таблицы, содержащие информацию по определенной теме, необходимо связать в единую структуру базы данных. Для связывания таблиц следует задать *ключевые поля*. Ключ состоит из одного или нескольких полей, значения которых *однозначно* определяют каждую запись в таблице. Наиболее подходящим в качестве ключевого поля является *«Счетчик»*, так как значения в данном поле являются уникальными (т. е. исключают повторы).

4. Откройте таблицу *Сотрудники* в режиме Конструктора.

5. Нажмите правой кнопкой мыши на поле *Код сотрудника* и в появившемся контекстном меню выберите команду *Ключевое поле*. Если в таблице необходимо установить несколько ключевых полей, то выделить их можно, удерживая клавишу *Ctrl*.

6. Для таблицы *Клиенты* установите ключевое поле *Код клиента*, а для таблицы *Заказы – Код заказа*.

7. Таблица *Заказы* содержит поля *Код сотрудника* и *Код клиента*. При их заполнении могут возникнуть некоторые трудности, так как не всегда удается запомнить все предприятия, с которыми работает фирма, и всех сотрудников с номером кода. Для удобства можно создать раскрывающиеся списки с помощью *Мастера подстановок*.

8. Откройте таблицу *Заказы* в режиме Конструктора.

9. Для поля *Код сотрудника* выберите тип данных *Мастер подстановок.*

10. В появившемся окне выберите команду «Объект «*столбец подстановки» будет использовать значения из таблицы или запроса*» и щелкните на кнопке *Далее*.

11. В списке таблиц выберите таблицу *Сотрудники* и щелкните на кнопке *Далее*.

12. В списке *Доступные поля* выберите поле *Код сотрудника* и щелкните на кнопке со стрелкой, чтобы ввести поле в список *Выбранные поля*. Таким же образом добавьте поля *Фамилия* и *Имя* и щелкните на кнопке *Далее*.

13. Выберите порядок сортировки списка по полю *Фамилия.*

14. В следующем диалоговом окне задайте необходимую ширину столбцов раскрывающегося списка.

15. Установите флажок *Скрыть ключевой столбе»* и нажмите кнопку *Далее*.

16. На последнем шаге *Мастера подстановок* замените при необходимости надпись для поля подстановок и щелкните на кнопке *Готово*.

17. Аналогичным образом создайте раскрывающийся список для поля *Код клиента*.

18. После создания ключевых полей можно приступить к созданию связей. Существует несколько типов отношений между таблицами:

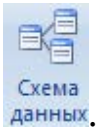
1) при отношении «*один-к-одному»* каждой записи ключевого поля в первой таблице соответствует только одна запись в связанном поле другой таблицы, и наоборот. Отношения такого типа используются не очень часто. Иногда их можно использовать для разделения таблиц, содержащих много полей, для отделения части таблицы по соображениям безопасности;

2) при отношении *«один-ко-многим»* каждой записи в первой таблице

соответствует несколько записей во второй, но запись во второй таблице не может иметь более одной связанной записи в первой таблице;

3) при отношении *«многие-ко-многим»* одной записи в первой таблице могут соответствовать несколько записей во второй таблице, а одной записи во второй таблице могут соответствовать несколько записей в первой.

19. Закройте все открытые таблицы, так как создавать или изменять связи между открытыми таблицами нельзя.

20. Выполните команду: вкладка ленты *Работа с базами данных* → кнопка 

.

21. Если ранее никаких связей между таблицами базы не было, то при открытии окна *Схема данных* одновременно открывается окно *Добавление таблицы*, в котором выберите таблицы *Сотрудники, Клиенты* и *Заказы*.

22. Если связи между таблицами уже были заданы, то для добавления в схему данных новой таблицы щелкните правой кнопкой мыши на схеме данных и в контекстном меню выберите пункт *Добавить таблицу*.

23. Установите связь между таблицами *Сотрудники* и *Заказы*, для этого выберите поле *Код сотрудника* в таблице *Сотрудники* и перенесите его на соответствующее поле в таблице *Заказы*.

24. После перетаскивания откроется диалоговое окно *Изменение связей* (рис. 25), в котором включите флажок *Обеспечение условия целостности*. Это позволит предотвратить случаи удаления записей из одной таблицы, при которых связанные с ними данные других таблиц останутся без связи.

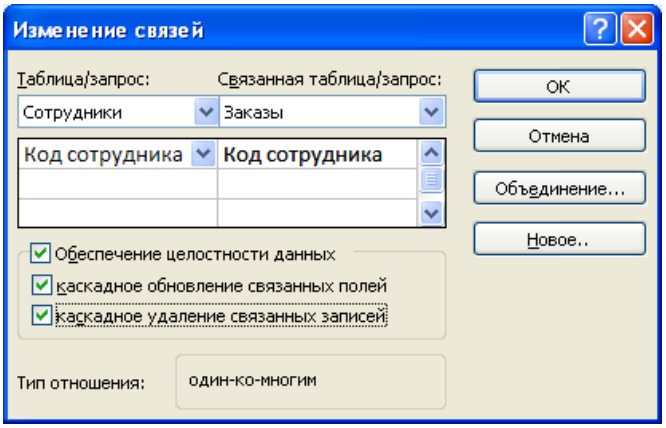


Рис. 25. Создание связи между таблицами

25. Флажки *Каскадное обновление связанных полей* и *Каскадное удаление связанных записей* обеспечивают одновременное обновление или удаление данных во всех подчиненных таблицах при их изменении в главной таблице.

26. Параметры связи можно изменить, нажав на кнопку *Объединение*.

27. После установления всех необходимых параметров нажмите кнопку *ОК*.

28. Связь между таблицами *Клиенты* и *Заказы* установите самостоятельно.

29. В результате должна получиться схема данных, представленная на рис. 26.

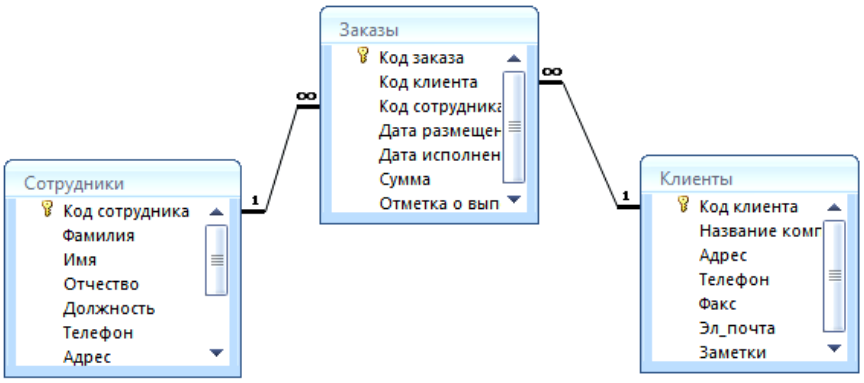


Рис. 26. Схема данных

В приведенном примере используются связи «один-ко-многим». На схеме данных они отображаются в виде соединительных линий со специальными значками около таблиц. Связь «один-ко-многим» помечается «1» вблизи главной таблицы (имеющей первичный ключ) и «∞» вблизи подчиненной таблицы (имеющей внешний ключ). Связь «один-к-одному» помечается двумя «1» (оба поля таблиц имеют первичные ключи). Неопределенная связь не имеет никаких знаков. Если установлено объединение, то его направление отмечается стрелкой на конце соединительной линии (ни одно из объединенных полей не является ключевым и не имеет уникального индекса).

30. В таблицу *Сотрудники* внесите данные о семи работниках.

31. В таблицу *Клиенты* внесите данные о десяти предприятиях, с которыми работает данная фирма.

32. В таблице *Заказы* оформите несколько заявок, поступивших на фирму.

4.3 Отбор данных с помощью запросов

Запросы являются основным средством просмотра, отбора, изменения и анализа информации, которая содержится в одной или нескольких таблицах базы данных.

Существуют различные виды запросов, но наиболее распространенными являются *запросы на выборку*, с них и начнем наше знакомство.

1. Откройте базу данных *«Фирма»*, созданную ранее.

2. Выполните команду: вкладка ленты *Создание* → *Мастер запросов* →*Простой запрос.*

3. В появившемся диалоговом окне (рис. 27) укажите таблицу *Сотрудники* и выберите поля *Фамилия, Имя, Телефон*. Нажмите кнопку *Далее*.

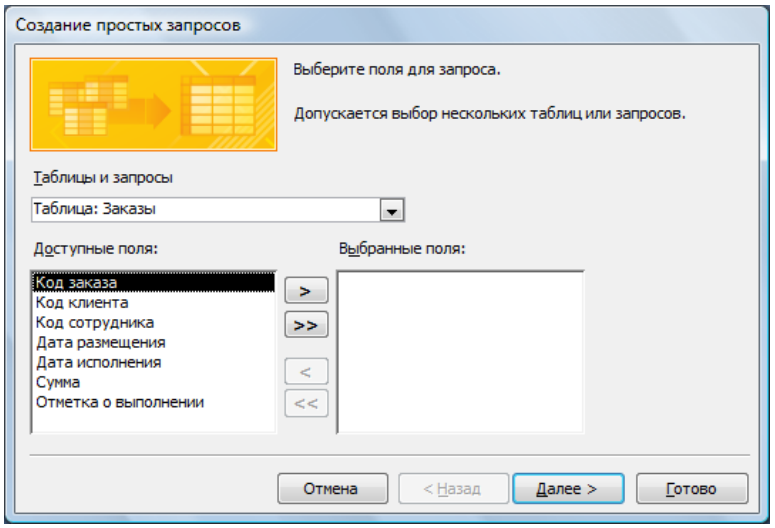


Рис. 27. Создание простого запроса

4. Введите имя запроса – *Телефоны* – и нажмите кнопку *Готово*. Перед вами появится запрос, в котором можно просмотреть телефоны сотрудников.

5. Следующий запрос попробуйте создать с помощью Конструктора, для этого выполните команду: вкладка ленты *Создание → Конструктор запросов.*

6. В диалоговом окне *Добавление таблиц* выберите таблицу *Клиенты* и щелкните на кнопке *Добавить,* а затем – на кнопке *Закрыть*.

7. Чтобы перенести нужные поля в бланк запроса, необходимо по ним дважды щелкнуть левой кнопкой мыши (рис. 28).

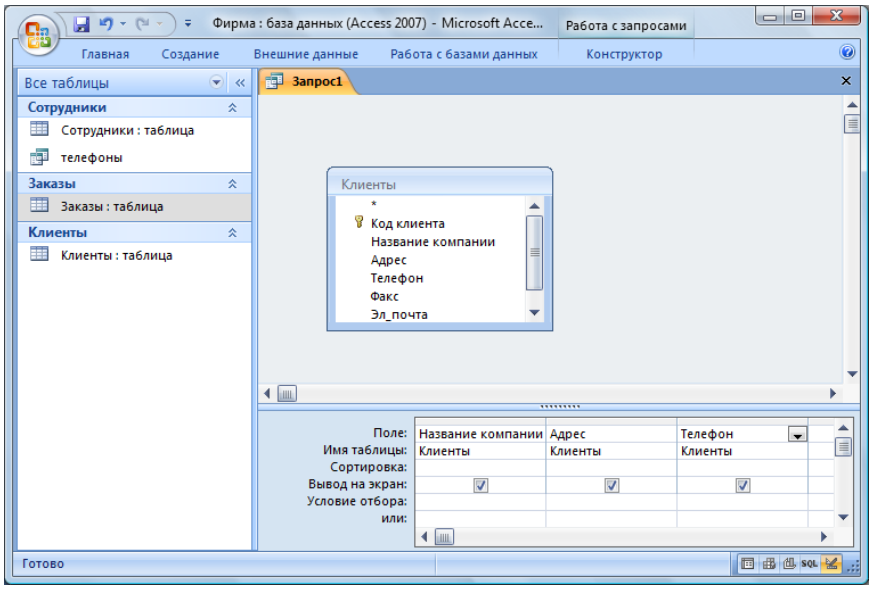


Рис. 28. Создание запроса в режиме Конструктора

8. Чтобы отсортировать записи в поле *Название компании* в алфавитном порядке, необходимо в раскрывающемся списке строки *Сортировка* выбрать пункт *по возрастанию*.

9. Сохраните запрос с именем *«Адреса клиентов»*.

10. Самостоятельно создайте запрос *«Дни рождения»*, в котором можно

будет просмотреть дни рождения сотрудников.

11. Допустим, мы хотим узнать, у кого из сотрудников день рождения в текущем месяце, например в апреле. Для этого откройте запрос в режиме Конструктора.

12. В строке *Условие отбора* для поля *«Дата рождения»* введите значение **.04.**. В данной записи * означают, что дата и год рождения могут быть любыми, а месяц 4-м (т. е. апрель). После этого окно запроса должно выглядеть так, как оно представлено на рис. 29.

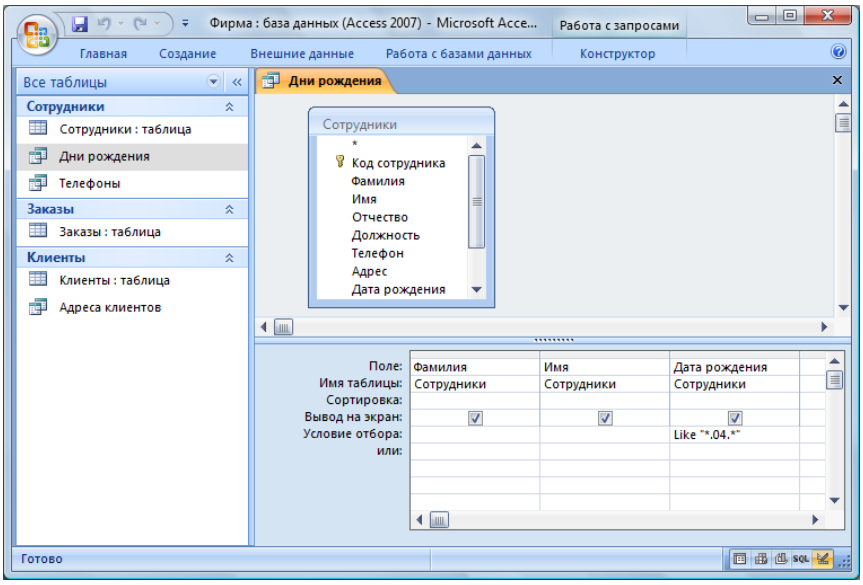


Рис. 29. Создание запроса

13. Закройте Конструктор и просмотрите полученный результат. Если в запросе *Дни рождения* нет ни одной записи, значит, в таблице *Сотрудники* нет ни одного человека, родившегося в апреле. Добавьте в таблицу *Сотрудники* несколько человек, родившихся в апреле, и посмотрите, как изменится запрос. Запросы автоматически обновляются при каждом открытии.

14. Если нам нужно узнать, кто из сотрудников родился в мае, то придется создать новый запрос или изменить условие в существующем запросе *Дни рождения*. Данная процедура является неудобной и занимает много времени.

Если приходится часто выполнять запрос, но каждый раз с новыми значениями условий используют *запрос с параметром*. При запуске такого запроса на экран выводится диалоговое окно для ввода значения в качестве условия отбора. Чтобы создать запрос с параметром, пользователю необходимо ввести текст сообщения в строке *Условие отбора* бланка запроса (рис. 30).

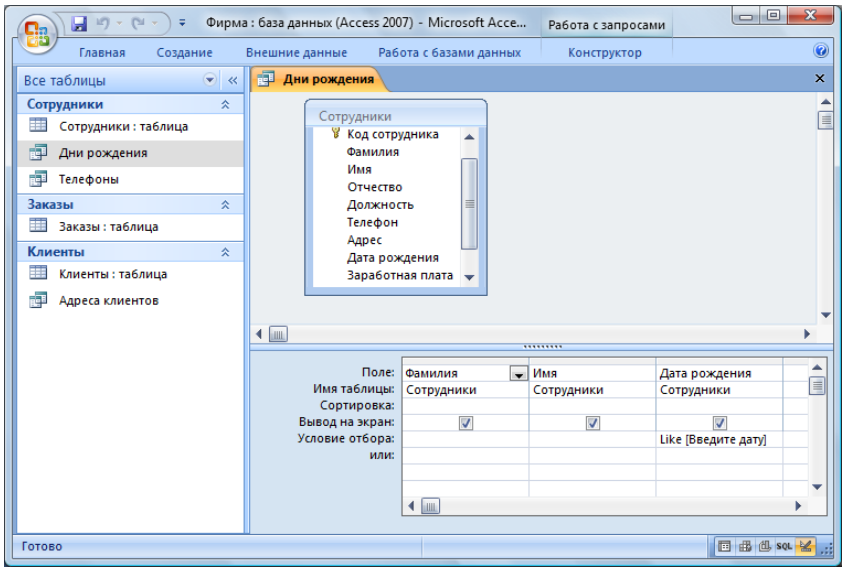


Рис. 30. Создание запроса с параметром

15. Запись *Like[Введите дату]* означает, что при открытии запроса появится диалоговое окно (рис. 31) с текстом *«Введите дату»* и полем для ввода условия отбора. Если ввести условие **.04.**, то в запросе появится список сотрудников, родившихся в апреле. Запустите запрос еще раз и введите значение **.05.**, посмотрите, как изменился запрос.

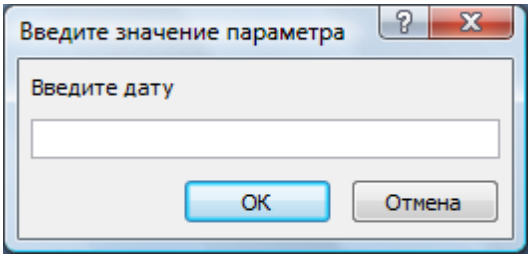


Рис. 31. Окно для ввода условия отбора

16. Измените запрос *«Телефоны»* так, чтобы при его запуске выводилось диалоговое окно с сообщением *«Введите фамилию»*. Поскольку в запросе нужно вывести конкретную фамилию, в условии отбора слово *Like* писать не надо.

17. Измените запрос *«Телефоны»* так, чтобы при его запуске запрашивались не только фамилия, но и имя сотрудника.

18. Самостоятельно создайте запрос *«Выполненные заказы»*, содержащий следующие сведения: фамилия и имя сотрудника, название компании, с которой он работает, отметка о выполнении и сумма заказа. Данные запроса возьмите из нескольких таблиц.

19. В условии отбора для логического поля *Отметка о выполнении* введите *Да*, чтобы в запросе отображались только выполненные заказы.

20. Сделайте так, чтобы столбец *Отметка о выполнении* не выводился на экран.

21. Создайте запрос *Сумма заказа*, в котором будут отображаться заказы на

сумму более *50 000 руб*.

22. Измените запрос, чтобы сумма заказа была от *20 000* до *50 000* руб.

Для данных запросов в условии отбора можно использовать операторы сравнения >, <, =, >=, <=, < > и логические операторы *And, Or, Not* и др.

23. Иногда в запросах требуется произвести некоторые вычисления, например посчитать подоходный налог *13 %* для каждой сделки. Для этого откройте запрос *Сумма заказа* в режиме Конструктора.

24. В пустом столбце бланка запроса щелкните правой кнопкой мыши на ячейке *Поле* и в появившемся контекстном меню выберите команду *Построить*. Перед вами появится окно *Построитель выражений* (рис. 8), который состоит из трех областей: поля выражения, кнопок операторов и элементов выражения. Сверху располагается поле выражения, в котором оно и создается.

Вводимые в это поле элементы выбираются в двух других областях окна Построителя.

25. В левом списке откройте папку *Запросы* и выделите запрос *Сумма заказа*. В среднем списке выделите поле *Сумма* и нажмите кнопку *Вставить*. Идентификатор этого поля появится в поле выражения Построителя.

26. Щелкните на кнопке * и введите *0,13* (см. рис. 32). Таким образом, мы посчитаем подоходный налог *13 %*.

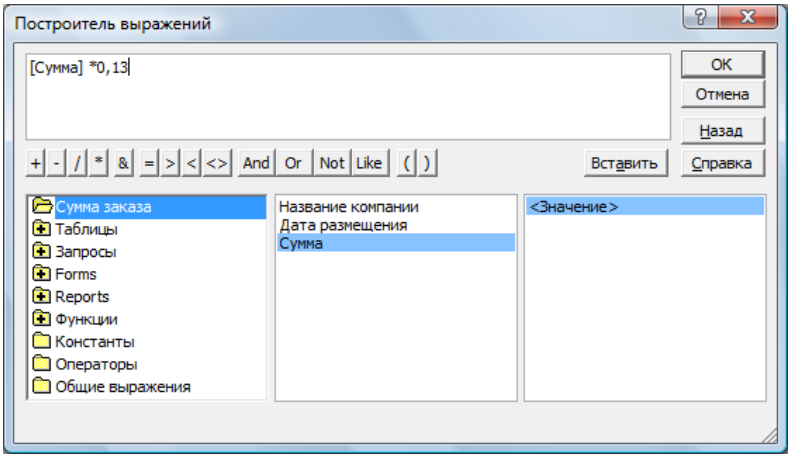


Рис. 32. Построитель выражений

27. Нажмите кнопку *ОК*, после чего в ячейке свойства *Поле* появится значение *«Выражение1: [Сумма]*0,13»*.

28. Замените *Выражение1* на *Налог* и закройте Конструктор.

29. Откройте запрос и посмотрите, что у вас получилось.

30. Используя *Построитель выражений*, добавьте в запрос *Сумма заказа* поле *Прибыль*, в котором будет вычисляться доход от заказа (т. е. сумма минус налог).

31. Создайте запрос *Менеджеры*, с помощью которого в таблице *Сотрудники* найдите всех менеджеров фирмы.

4.4 Использование форм в базе данных

Формы – это объекты базы данных, предназначенные для просмотра данных из таблиц и запросов, для ввода данных в базу, корректирования существующих данных и выполнения заданных действий. Форма может содержать графики, рисунки и другие внедренные объекты.

Можно вносить данные в таблицы и без помощи каких-либо форм. Но существует несколько причин, которые делают формы незаменимым средством ввода данных в базу:

1) при работе с формами ограничен доступ к таблицам (самому ценному в базе данных);

2) разные люди могут иметь разные права доступа к информации, хранящейся в базе. Для ввода данных им предоставляются разные формы, хотя данные из форм могут поступать в одну таблицу;

3) вводить данные в форму легче, чем в таблицу, и удобнее, так как в окне формы отображается, как правило, одна запись таблицы;

4) в большинстве случаев информация для баз данных берется из бумажных бланков (анкет, счетов, накладных, справок и т. д.). Экранные формы можно сделать точной копией бумажных бланков, благодаря этому уменьшается количество ошибок при вводе и снижается утомляемость персонала.

Создавать формы можно на основе нескольких таблиц или запросов с помощью Мастера, используя средство автоформы, «вручную» в режиме Конструктора, сохраняя таблицу или запрос как форму. Созданную любым способом форму можно затем изменять в режиме Конструктора. Рассмотрим некоторые из перечисленных способов.

1. Выполните команду: вкладка ленты Создание → панель инструментов *Формы → Другие формы → Мастер форм.*

2. В диалоговом окне *Создание форм* выберите таблицы (запросы) и поля, которые будут помещены в форму. Щелкните по кнопке *Далее*.

3. В следующих диалоговых окнах мастера выберите внешний вид формы, стиль, задайте имя формы. Щелкните по кнопке *Готово*.

4. С помощью Мастера создайте формы *Сотрудники*, *Клиенты*, *Заказы*, *Менеджеры.*

5. Откройте форму *Сотрудники* в режиме Конструктора. Этот режим предназначен для создания и редактирования форм.

6. Разместите элементы в удобном для вас порядке, измените размер и цвет текста.

7. В заголовок формы добавьте текст *Сотрудники фирмы*.

8. В примечание формы добавьте объект *Кнопка* (вкладка ленты *Конструктор* → панель инструментов *Элементы управления*).

9. После того как вы «нарисуете» кнопку указателем, на экране появится диалоговое окно *Создание кнопок* (рис. 33).

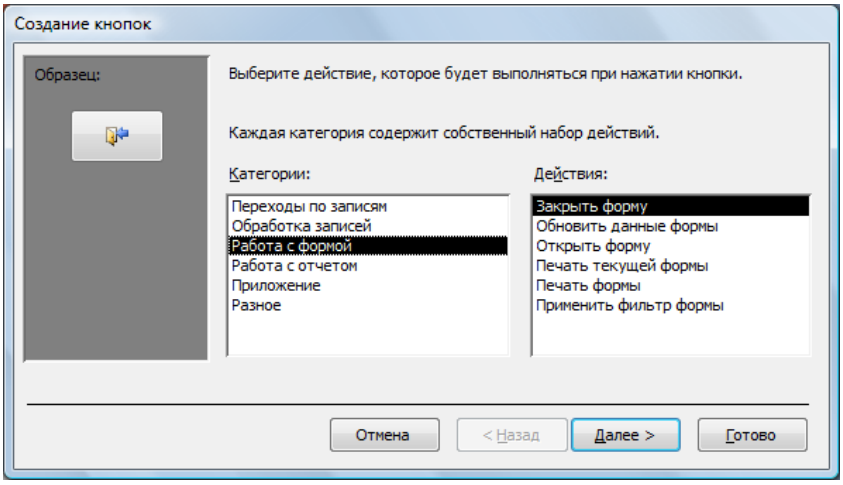


Рис. 33. Создание кнопок на форме

10. В категории *Работа с формой* выберите действие *Закрытие формы* и нажмите кнопку *Далее*.

11. Выберите рисунок или текст, который будет размещаться на кнопке.

12. В последнем диалоговом окне *Мастера кнопок* задайте имя кнопки и нажмите *Готово*.

13. *Мастер кнопок* написал для данной кнопки процедуру на языке Microsoft Visual Basic. Просмотреть процедуру обработки события можно с помощью команды *Обработка событий* контекстного меню кнопки.

14. Самостоятельно создайте кнопки *Выход из приложения*, *Поиск записи*, *Удаление записи*.

15. Иногда на форме требуется разместить несколько страниц, содержащих данные из различных источников, справочную или вспомогательную информацию. Для этой цели можно использовать набор вкладок.

16. Создайте пустую форму.

17. Для добавления к форме набора вкладок щелкните по кнопке *Вкладка* на панели инструментов *Элементы управления*. Сначала добавятся только две вкладки с формальными именами *Вкладка 1* и *Вкладка 2*.

18. Добавьте еще одну вкладку.

19. Переименуйте ярлычки вкладок так, чтобы на них отображались названия данных, которые будут в них располагаться: *Сотрудники, Менеджеры, Помощь*.

20. Перейдите на вкладку *Сотрудники* и перетащите на нее мышкой из базы данных форму *Сотрудники*.

21. Аналогичным образом поместите форму *Менеджеры* на вкладку *Менеджеры*.

22. На вкладку *Помощь* поместите некоторые советы по работе с базой

данных.

23. Данную форму сохраните с именем *Сотрудники фирмы.*

24. В *Microsoft Access* можно создавать кнопочные формы. Они содержат только кнопки и предназначены для выбора основных действий в базе данных.

Для создания кнопочной формы необходимо на вкладке ленты *Работа с базами данных* выбрать команду *Диспетчер кнопочных форм*.

25. Если кнопочной формы в базе данных нет, то будет выведен запрос на подтверждение ее создания. Нажмите *Да* в диалоговом окне подтверждения.

26. Перед вами появится *Диспетчер кнопочных форм*, в котором щелкните по кнопке *Создать*.

27. В диалоговом окне *Создание* (рис. 34) введите имя новой кнопочной формы и нажмите *ОК*.

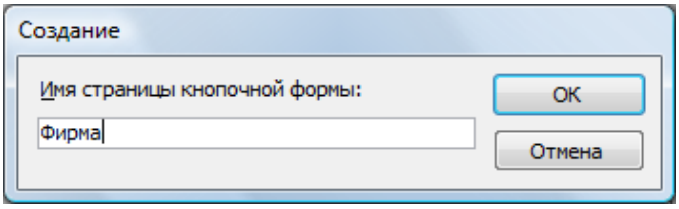


Рис. 34. Задание имени кнопочной формы

28. Имя новой кнопочной формы добавится в список *Страницы кнопочной формы* окна *Диспетчер кнопочных форм* (рис. 35). Выделите имя новой кнопочной формы и щелкните по кнопке *Изменить*.

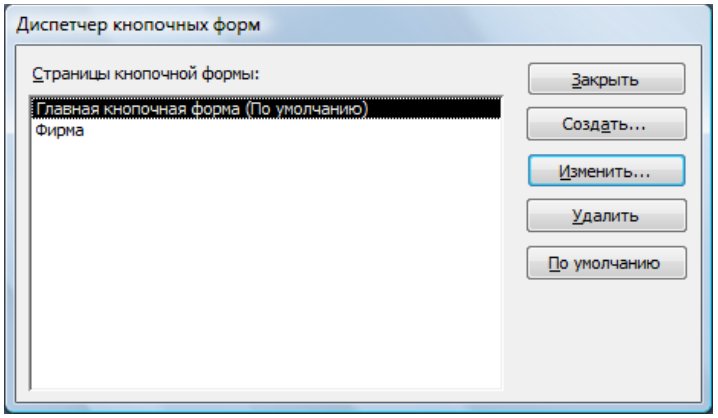


Рис. 35. Диспетчер кнопочных форм

29. В диалоговом окне *Изменение страницы кнопочной формы* щелкните по кнопке *Создать*. Появится диалоговое окно *Изменение элемента кнопочной формы* (рис. 36).

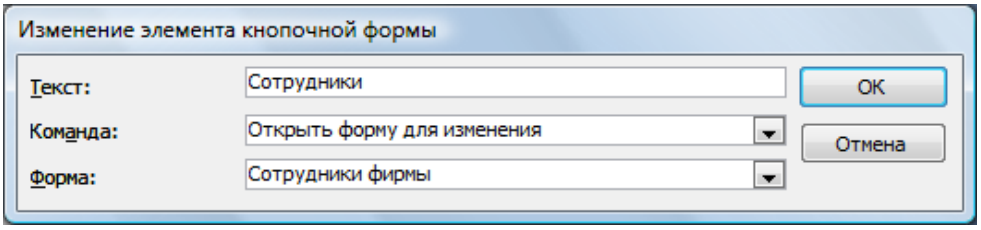


Рис. 36. Создание кнопок на форме

30. В поле *Текст* введите текст подписи для первой кнопки кнопочной формы, а затем выберите команду из раскрывающегося списка в поле *Команда*.

В поле *Форма* выберите форму, для которой будет выполняться данная команда.

31. Аналогичным образом добавьте кнопки *Клиенты*, *Заказы*, *Выход.*

32. В диалоговом окне *Диспетчер кнопочных форм* выберите имя вашей кнопочной формы и щелкните по кнопке *По умолчанию*. Рядом с названием кнопочной формы появится надпись *«(по умолчанию)»*.

33. Чтобы закончить создание кнопочной формы, щелкните по кнопке *Закрыть*.

34. В результате должна получиться форма, представленная на рис. 37.

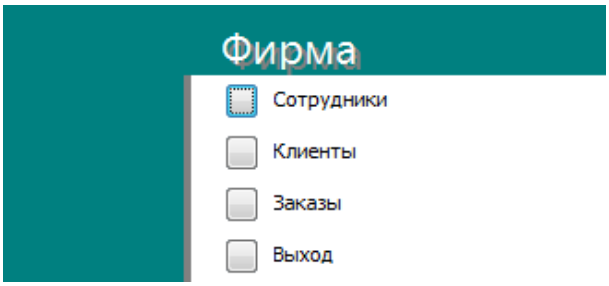


Рис. 37. Главная кнопочная форма

35. Добавьте в форму какой-нибудь рисунок.

36. Для того чтобы главная кнопочная форма появлялась на экране при запуске приложения, необходимо в главном меню нажать на кнопку *Параметры Access*. Для текущей базы данных установите форму просмотра – «кнопочная форма».

4.5 Создание отчетов

Отчеты предназначены для вывода информации на печать. Часто данные в них располагаются в табличной форме. В отличие от распечаток таблиц или запросов отчет дает более широкие возможности сортировки и группировки данных, он предоставляет возможность добавлять итоговые значения, а также поясняющие надписи, колонтитулы, номера страниц, стили и различные графические элементы.

Создавать отчеты в базе данных Access можно несколькими способами:

1) с помощью Мастера отчетов;

2) на основе таблиц или запросов;

3) в режиме Конструктора.

1. В окне базы данных выполните команду: вкладка ленты *Создание* →панель инструментов *Отчеты* → *Мастер отчетов*.

2. Выберите из списка таблицу (или запрос), которая будет использована как источник данных (например, запрос *Адреса клиентов*).

3. В появившемся диалоговом окне *Создание отчетов* (рис. 38)

переместите все доступные поля в область «выбранные поля».

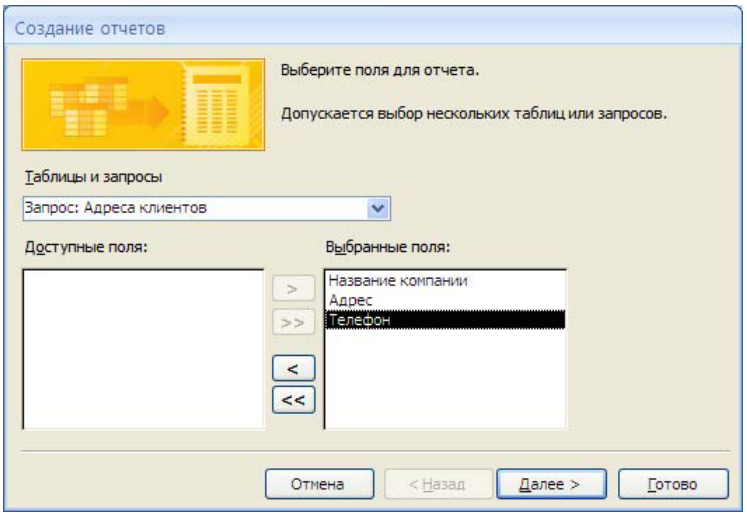


Рис. 38. Мастер отчетов

4. С помощью *Мастера отчетов* создайте отчет *Дни рождения*. В качестве источника данных используйте таблицу *Сотрудники*.

5. Если требуется напечатать почтовые наклейки, Access предоставляет такую возможность. Для этого выделите таблицу *Клиенты* и выполните команду: вкладка ленты *Создание* → панель инструментов *Отчеты* → *Наклейки.*

6. В появившемся диалоговом окне (рис. 39) укажите размер наклейки, систему единиц, тип наклейки и нажмите кнопку *Далее*.

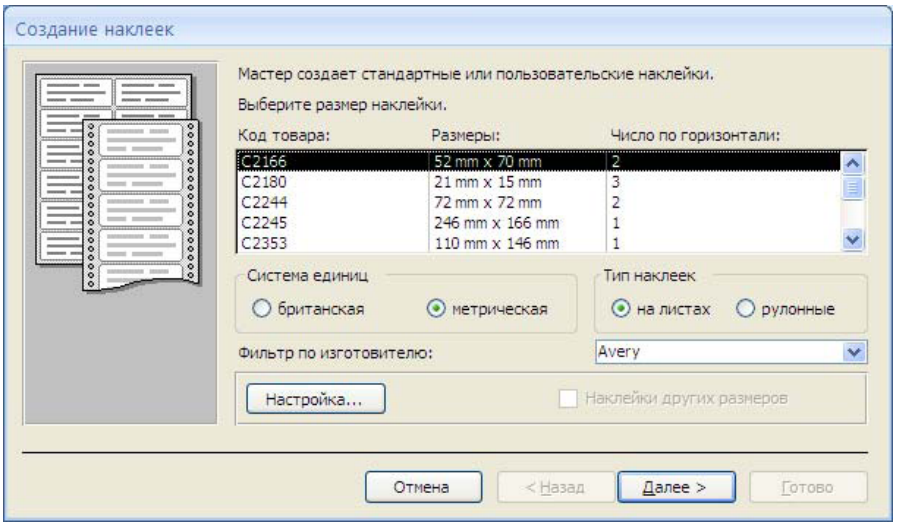


Рис. 39. Диалоговое окно Создание наклеек

7. На следующем шаге создания отчета установите шрифт, размер, цвет текста и начертание. Нажмите кнопку *Далее*.

8. Выберите поля, которые будут размещаться на наклейке. Например, *Название компании*, *Адрес*, *Телефон* и *Факс*. Если на каждой наклейке требуется вывести определенный текст, то введите его в прототип наклейки.

9. При необходимости измените название отчета с наклейками и нажмите кнопку *Готово*.

10. Иногда в отчетах требуется вычислять итоговые значения, среднее, минимальное или максимальное значения, а также проценты. Для этого запустите *Мастер отчетов* и в качестве источника данных укажите запрос *Сумма заказа*.

11. В диалоговом окне *Мастера*, в котором задается порядок сортировки записей, нажмите кнопку *Итоги* (рис. 40).

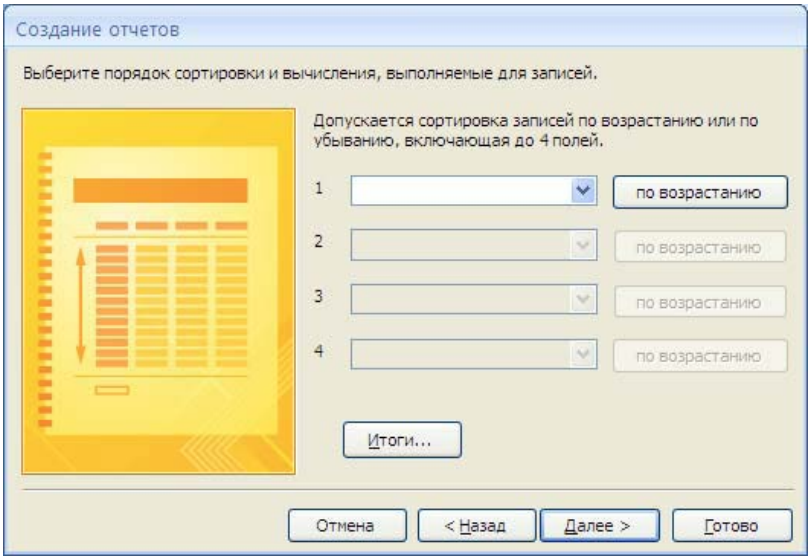


Рис. 40. Вычисление итоговых значений в отчетах

12. В диалоговом окне *Итоги* (рис. 41) для полей *Сумма* и *Налог* установите флажки в столбце *sum*, чтобы посчитать итоговую сумму.

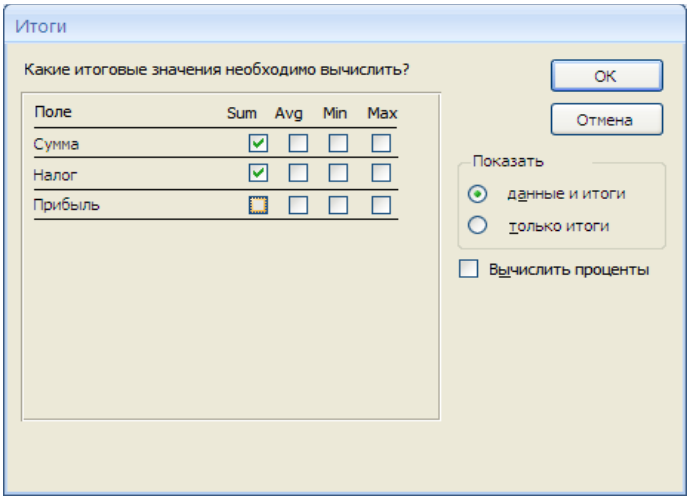


Рис. 41. Вычисление итоговых значаний суммы

13. Далее выполните все шаги Мастера и нажмите кнопку *Готово*.

14. Создайте отчет *Дни рождения*, используя в качестве источника данных запрос *Дни рождения*.

15. Составьте отчет *Выполненные заказы*, в котором будут данные о компании и сумме заказа. Вычислите итоговую сумму, среднее значение (Avg) и максимальную сумму для каждой фирмы.

ТЕМА 6. АЛГОРИТМИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

План:

1. Понятие алгоритма.
2. Способы представления алгоритма.
3. Базовые алгоритмические конструкции.
4. Оценка эффективности алгоритмов.

Вопрос 1. Понятие алгоритма

Алгоритм – это понятное и точное представление исполнителю совершить последовательность действий, направленных на решение поставленной задачи или достижение указанной цели.

В IX веке среднеазиатский математик аль-Хорезми разработал правила арифметических действий над десятичными числами. Совокупность этих правил в Европе стали называть «алгоритм». Впоследствии это слово расширило свое значение: алгоритмом стали называть любую последовательность действий (не только арифметических), которая приводит к решению той или иной задачи. Можно сказать, что понятие вышло за рамки математики и стало применяться в самых различных областях.

Можно выделить три разновидности алгоритмов: вычислительные, информационные и управляющие. Первые, как правило, работают с простыми видами данных (числа, вектора, матрицы), но зато процесс вычисления может быть длинным и сложным. Информационные алгоритмы, напротив, реализуют сравнительно небольшие процедуры обработки (например, поиск элементов, удовлетворяющих определенному признаку), но для больших объемов информации. Наконец, управляющие алгоритмы непрерывно анализируют информацию, поступающую от тех или иных источников, и выдают результирующие сигналы, управляющие работой тех или иных устройств. Для этого вида алгоритмов очень существенную роль играет их быстродействие, так как управляющие сигналы всегда должны появляться в нужный момент времени.

Для того чтобы произвольное описание последовательности действий было алгоритмом, оно должно обладать следующими свойствами:

1. Дискретность. Процесс решения задачи должен быть разбит на последовательность отдельных шагов, каждый из которых называется командной. Примером команд могут служить пункты инструкции, нажатие на одну их кнопок пульта управления, рисование графического примитива (линии, дуги и т.п.), оператор языка программирования. Наиболее существенным здесь является тот факт, что алгоритм есть последовательность четко выделенных пунктов –

такие «прерывные» объекты в науке принято называть дискретными.

2. Понятность. Каждая команда алгоритма должна быть понятна тому, кто исполняет алгоритм; в противном случае эта команда и, следовательно, весь алгоритм в целом не могут быть выполнены. Данное требование можно сформулировать более просто и конкретно. Составим полный список команд, который умеет делать исполнитель алгоритма, и назовем его системой команд исполнителя. Основным из таких универсальныхисполнителей является ЭВМ. Это связано с тем, что любой алгоритм, составленный для ЭВМ транслируется в ее системе команд исполнителяи, таким образом, становится доступным для исполнения.

3. Определенность (детерминированность). Команды, образующие алгоритм, должны быть предельно четкими и однозначными. Их результат не может зависеть от какой-либо дополнительной информации извне алгоритма. Сколько бы раз вы не запускали программу, для одних и тех же исходных данных всегда будет получаться один и тот же результат. Определенность также предполагает, что данные, необходимые для выполнения очередной команды алгоритма, получены на одном из предыдущих шагов алгоритма.

4. Корректность. Любой алгоритм создан для решения той или иной задачи, поэтому необходима уверенность, что это решение будет правильным для любых допустимых исходных данных. В связи с этим свойством большое значение имеет тщательное тестирование алгоритма перед его использованием. Грамотная и всесторонняя отладка для сложных алгоритмов часто требует значительно больших усилий, чем разработка алгоритмов. При этом важно не столько количество проверенных сочетаний входных данных, сколько количество их типов.

5. Массовость. Алгоритм имеет смысл разрабатывать только в том случае, когда он будет применяться многократно для различных наборов исходных данных. Например, если составляется алгоритм обработки текстов, то вряд ли целесообразно ограничить его возможности только русскими буквами – стоит предусмотреть также латинский алфавит, цифры, знаки препинания и т.д. Тем более что обобщение особых трудностей не вызывает.

6. Результативность (конечность). Алгоритм должен приводить к решению задачи за конечное количество шагов.

Процесс разработки алгоритма называется **алгоритмизацией**. Процесс разработки алгоритма сложной задачи требует квалификации, четкого и полного понимания задачи. Сущность алгоритмизациивычислительного процесса проявляется в следующих действиях:

- выделение законченных частей вычислительного процесса;
- формальная запись каждого из них;
- назначение определенного порядка выполнения выделенных частей;

▪ проверка правильности выбранного алгоритма.

Из возможности формального исполнения алгоритма следует, что поскольку осознавать содержание алгоритма не требуется, его исполнение вполне можно доверить автомату или ЭВМ. Таким образом, составление алгоритма является обязательным этапом автоматизации любого процесса. Как только разработан алгоритм, машина может исполнять его лучше человека – быстрее и, что очень важно, не ошибаясь.

Вопрос 2. Способы представления алгоритма

Основными способами записи алгоритмов являются:

1. Словесный способ. Описание алгоритма состоит из словесного перечня действий. Недостатком такого представления является отсутствие четкой формализации и наглядности выполнения процесса. Но таким способом можно описывать алгоритмы с любой степенью детализации.

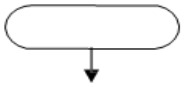
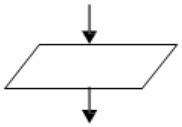
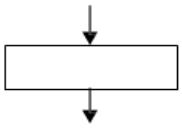
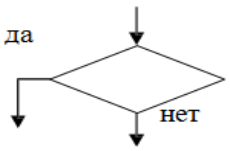
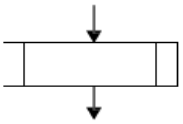
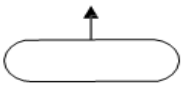
2. Словесно-формальный способ основан на знании инструкций о выполнении конкретных действий вчеткой последовательности, в сочетании со словесными пояснениями. Этот способ более компактен, чем словесный, но не является строго формальным.

3. Табличный способ. Алгоритм задается в виде таблиц и расчетных форм. Этот способ наиболее часто применяют в экономических расчетах. Исходные данные и результаты вносятся в заголовки столбцов таблицы.

4. Графический способ (блок-схема). Каждый этап отображается в виде геометрических фигур – блоков, форма которых зависит от выполняемых операций. Блок может иметь имя (метку). Линия соединения блоков показывает направление процесса обработки данных. Каждое направление называется ветвью. Перечень блоков, их наименование, функции, формы, размеры, взаиморасположение определяются ГОСТ 19.003-80. Условное обозначение блоков и их описание приведены в табл. 7.

5. Операторный способ. При использовании этого способа вычислительный процесс представляется в виде последовательности символов (операторов). Они обозначают группы стандартных или нестандартных операций, реализующих законченную процедуру с указанием связи между отдельными операторами. Этот способ значительно упрощает составление программы для компьютера, при этом операторы выражают соответствующие команды. Недостатком такого метода является его малая наглядность.

Таблица 7

№	Наименование блока	Графическое представление блока	Функция блока
1	2	3	4
1	Начало алгоритма		Начало процесса обработки
2	Ввод данных, вывод результата		Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод).
3	Вычисление		Выполнение операции или группы операций, в результате которых изменяются значение, форма представления или расположение данных
4	Проверка условия		Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий
5	Предопределенный (заранее описанный) процесс, модуль		Использование заранее сознанных или отдельно описанных алгоритмов (модулей)
6	Соединитель		Указание связи между прерванными линиями потока обработки данных
7	Конец алгоритма		Конец процесса обработки

Вопрос 3. Базовые алгоритмические конструкции

Для выполнения последовательностей действий в алгоритме разработаны базовые алгоритмические конструкции в виде определенного набора блоков и стандартных средств их соединения. Т.е. все вычислительные процессы можно разделить на три основных категории: линейные, разветвляющиеся и циклические.

Линейный вычислительный процесс (линейный алгоритм, следование) – это процесс, в котором все блоки выполняются один за другим в естественном порядке его следствия.

Примером такого алгоритма может служить процесс вычисления функции по формуле $y = \sin(ax+b)/(ax+b)^2$, который представлен в виде блок-схемы на рис. 42.

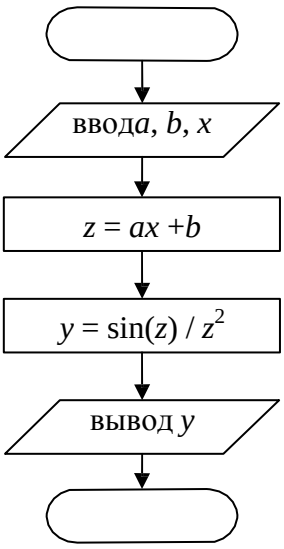


Рис. 42. Пример линейного алгоритма

Разветвляющийся вычислительный процесс или **алгоритм ветвления**. Такие процессы наиболее часто встречаются на практике, когда в зависимости от промежуточных результатов необходимо выполнять вычисления по одним или другим формулам, т.е. в зависимости от выполнения какого-либо логического условия, процесс вычислений должен идти по одной или другой ветви.

В качестве условия в разветвляющемся алгоритме может быть использовано любое понятное исполнителю утверждение, которое может соблюдаться (быть истинным – да) или не соблюдаться (быть ложным – нет). Такое утверждение может быть выражено как словами, так и формулой. Таким образом, алгоритм ветвления состоит из условия и двух команд.

Ветвления играют в алгоритмах очень большую роль, поскольку предусматривают корректную реакцию на самые разнообразные ситуации, возникающие в процессе обработки информации. Благодаря этой структуре алгоритм приобретает способность выбирать один из существующих вариантов работы, наиболее подходящих к сложнейших в данный момент ситуации. В частном случае речь может идти о выполнении или игнорировании при определенных условиях того или иного участка алгоритма.

Значение ветвления в современном программном обеспечении трудно переоценить. Достаточно вспомнить элементы управления, такие, как меню, флажки проверки или списки. Именно они дают возможность пользователю чувствовать себя за компьютером свободно и комфортно и выбирать те режимы работы, которые ему нужны.

Примеромразветвляющегося алгоритма может служить процесс выбора наименьшего из двух введенных произвольных чисел A и B (рис. 43).

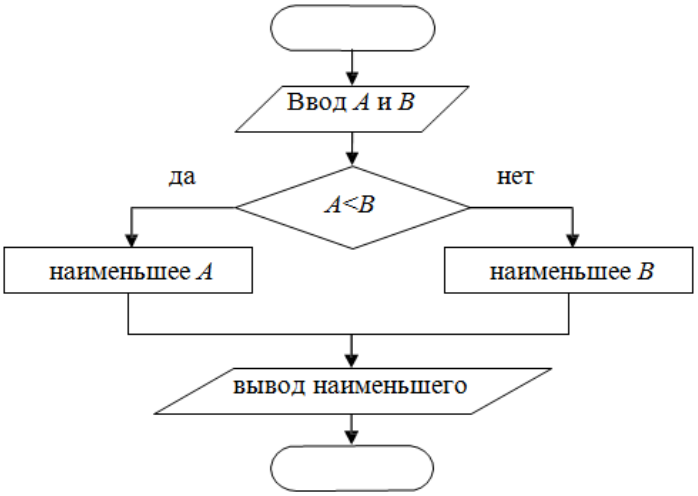


Рис. 43. Пример разветвляющегося алгоритма

В табл. 8 приведены полные формы ветвления в различных алгоритмических языках.

Таблица 8

Запись оператора разветвления на языках программирования

Язык программирования	Запись оператора
QBasic	IF<логическое выражение>THEN операторы ELSE операторы ENDIF
Pascal	IF< логическое выражение >T
C++	if (<логическое выражение >) оператор1; else оператор2;

Достаточно часто при организации алгоритма решение задачи необходимо одну и ту же определенную последовательность команд выполнить несколько раз подряд. Конечно, самый простой способ – записать эти команды несколько раз друг за другом, и необходимое повторение действий будит организовано. Но часто бывает, что количество команд, которые исполняются несколько раз, слишком велико. Решить эти проблемы можно, если использовать алгоритмическую структуру «цикл».

Циклический вычислительный процесс (циклическийалгоритм) – процесс, в котором вычисление сводится к многократному вычислению по одним и тем же математическим зависимостям (формулам) при различных значениях входящих в них величин. Многократно повторяющиеся участки такого вычислительного

Командой повторения, или **циклом**, называется такая форма организации

действий в алгоритме, при которой выполнение одной и той же последовательности команд повторяется до тех пор, пока истинно некоторое логическое выражение.

Примером циклического алгоритма может служить процесс расчета суммы S введенных произвольных чисел x, пока ее значение не превысит 100 (рис. 44).

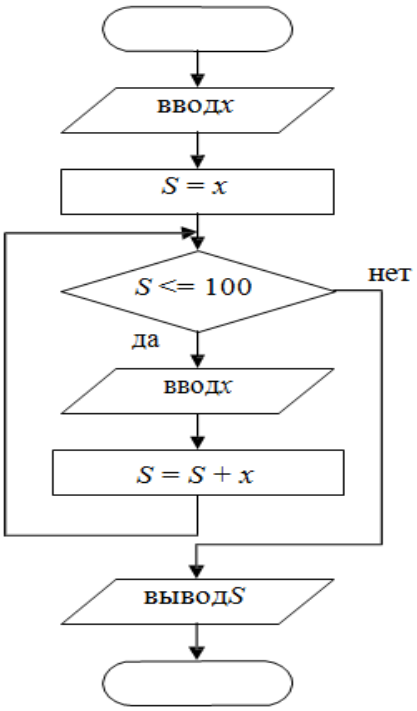


Рис. 44. Пример циклическогоалгоритма

Для организации цикла необходимо выполнить следующие действия:

- перед началом цикла задать начальное значение параметров (переменных, используемых в логическом выражении, отвечающем за продолжение или завершение цикла) – переменной цикла;
- выполнить операторы внутри цикла – тело цикла;
- внутри цикла изменять переменную (или переменные), которая сменит значение логического выражения, за счет которого продолжается цикл, на противоположное (для того чтобы цикл в определенный момент завершился) – изменение переменной цикла на заданный шаг;
- вычислить логическое выражение – проверить условие продолжения или окончания цикла;
- управлять циклом, т.е. переходить к его началу, если он не закончен, или выходить из цикла в противоположном случае.

Различают циклы с известным числом повторений (цикл с параметром) и итерационные (с пред- и пост- условием).

Цикл с предусловием или начальной проверкой условия предполагает, что

тело цикла расположено после проверки условия. Возможно, что при определенных условиях тело цикла не выполниться ни разу. При использовании цикла с предусловием выполняются следующие шаги:

1) вычисляется значение логического выражения;

2) если значение логического выражения «истина», то происходит переход к следующему пункту 3, если – «ложь», то происходит переход к пункту 5;

3) выполняется тело цикла;

4) переход к пункту 1;

5) конец цикла.

Цикл с постусловием или проверкой условия после выполненных вычислений предполагает, что тело цикла будет выполняться, по крайней мере, один раз, и будет повторяться до тех пор, пока условие не станет истинным. При использовании цикла с постусловием выполняются следующие шаги:

1) выполняется тело цикла;

2) вычисляется значение логического выражения;

3) если значение логического выражения «ложь», то происходит переход к пункту 1, если – «ложь», то происходит переход к следующему пункту 4;

4) конец цикла.

Третий тип циклов, в которых тело цикла выполняется заданное число раз (то есть еще до начала выполнения цикла точно известно, сколько раз он будет выполнен), реализуется с помощью счетчика. Такой цикл называется циклом с параметром. При использовании цикла с параметром выполняются следующие шаги:

1) вычисляются значения выражений, определяющие начальное значение параметра цикла;

2) параметру цикла присваивается начальное значение;

3) параметр цикла сравнивается с конечным значением;

4) если параметр цикла превосходит (при положительном шаге) конечное значение параметра цикла (или, наоборот, меньше конечного значение параметра цикла при отрицательном шаге), то происходит переход к пункту 8, иначе к следующему пункту 5;

5) выполняется тело цикла;

6) параметр цикла автоматически увеличивается на значение шага;

7) переход к пункту 3;

8) конец цикла.

Циклы с предусловием и постусловием в большинстве случаев (за исклю-

чением отдельных реализаций алгоритмических языков) являются более универсальными по сравнению с циклом с параметром, поскольку в последнем требуется заранее указать число повторений, в то время как в первых двух это не требуется. Цикл с параметром в любом случае может быть преобразован к циклу с пред- или постусловием.

Примеры записи циклов для различных языков программирования представлены в табл. 16. Принятые в таблице обозначения ЛВ – логическое выражение, ПЦ – переменная цикла, НЗ – начальное значение, КЗ – конечное значение, Ш – шаг изменения переменной цикла.

Таблица 9

Запись цикла на языках программирования

Язык программирования		
QBasic	*Pascal*	*C++*
цикл с предусловием		
while<ЛВ> операторы wend;	while<ЛВ>do оператор;	while (<ЛВ>) оператор;
цикл с постусловием		
loop операторы until<ЛВ>	repeat операторы until<ЛВ>;	do оператор while (<ЛВ>);
цикл с параметром		
for <ПЦ> = <НЗ> to <КЗ> step<Ш>операторы next<ПЦ>	for <ПЦ> = <НЗ> to (down-to) <КЗ>do оператор;	for (<ПЦ> = <НЗ>; <НЗ><=<КЗ>; <ПЦ> = <ПЦ>+<Ш>) оператор;

Вопрос 4. Оценка эффективности алгоритмов

Одну и ту же задачу могут решать много алгоритмов. Эффективность работы каждого из них описывается разнообразными характеристиками. Прежде чем анализировать эффективность алгоритма, нужно доказать, что данный алгоритм правильно решает задачу, то можно посмотреть, насколько это решение эффективно.

При анализе алгоритма определяется количество «времени», необходимое для его выполнения. Это не реальное число секунд или других промежутков времени, а приблизительное число операций, выполняемых алгоритмом. Число операций и измеряет относительное время выполнения алгоритма. Таким образом, иногда «временем» называют вычислительную сложность алгоритма. Фактическое количество секунд, требуемое для выполнения алгоритма на компьютере, непригодно для анализа, так как обычно интересует только относительная эффективность алгоритма, решающего конкретную задачу. Действительно, время, требуемое на решение задачи, не очень хороший способ измерять эф-

фективность алгоритма, потому что алгоритм не становится лучше, если его исполнять на более медленном ПК.

На самом деле фактическое количество операций алгоритма на тех или иных входных данных не представляет большого интереса и не очень много сообщает об алгоритме. Реально определяется зависимость числа операций конкретного алгоритма на тех или иных входных данных. Можно сравнить два алгоритма по скорости роста числа операций. Именно скорость роста играет ключевую роль, поскольку при небольшом размере входных данных алгоритм *A* может требовать меньшего количества операций, чем алгоритм *B*, но при росте объема входных данных ситуация может поменяться на противоположную.

Два самых больших класса алгоритмов – это алгоритмы с повторением и рекурсивные алгоритмы. В основе алгоритмов с повторением лежат циклы и условные выражения; для анализа алгоритмов требуется оценить число операций, выполняемых внутри цикла, и число итераций цикла. Рекурсивные алгоритмы разбивают большую задачу на фрагменты и применяются к каждому фрагменту по отдельности. В процесс решения большой задачи путем деления ее на меньшие создаются небольшие, простые и понятные алгоритмы. Анализ рекурсивного алгоритма требует подсчета количества операций, необходимых для разбиения задачи на части, выполнения алгоритма на каждойиз частей и объединения отдельных результатов для решения задачи в целом. Объединяя эту информацию и информацию о числе частей и их размере, можно вывести рекуррентное соотношение для сложности алгоритма. Полученному рекуррентному соотношению можно придать замкнутый вид, затем сравнивать результат с другими выражениями.

ТЕМА 7. ТЕХНОЛОГИИ И ИНСТРУМЕНТЫ ПРОГРАММИРОВАНИЯ

План:

1. Язык программирования.
2. Основные этапы разработки программ.
3. Технология программирования.
4. Использование языка программирования *Visual Basic.*
5. Технология офисного программирования в среде VBA.

Вопрос 1. Язык программирования

Язык программирования предназначен для описания алгоритмов обработки данных на ЭВМ. Он занимает промежуточное положение между естественными и формализованными языками.

Язык программирования – это формализованный набор правил и команд для описания алгоритмов решения задачи на компьютере.

Существуют различные способы составления команд для выполнения необходимых вычислений на компьютере. **Компьютерная программа** – это последовательность директив на машинно-ориентированном языке, выполняемая компьютером. От выбора языка программирования зависит, как будут использоваться ресурсы компьютера, а также сложность и скорость выполнения разработанных программ. Компьютерную программу принято считать результатом программирования. **Программирование** – это совокупность процессов, связанных с разработкой программ и их реализацией.

Одним из базовых принципов архитектуры современных компьютеров является двоичный принцип описания любой используемой информации, причем программа обработки также представляет собой двоичный код. Но для человека программировать в двоичных кодах неудобно и малоэффективно, поэтому используя языки программирования проще описать алгоритм решения задачи. Переход от языковых конструкций к машинным командам осуществляет специальная программа – **транслятор языка**.

Любой алгоритм – это последовательность команд, выполнив которые можно за конечное число шагов получить результат. Степенью детализации команд обычно определяется уровень языка программирования, т.е. степень его близости ЭВМ или человеку – чем меньше детализация, тем выше уровень языка.

Языки программирования можно разбить на следующие уровни:

- 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.

Язык ассемблера 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.

Программы, написанные на языке ассемблера, требуют значительно меньшего объема памяти и времени выполнения. Знание программистом языка ассемблера и машинного кода дает ему понимание архитектуры машины. Несмотря на то, что большинство специалистов в области программного обеспечения разрабатывают программы на языках высокого уровня, таких, как *Object Pascal* или *C*, наиболее мощное и эффективное программное обеспечение полностью или частично написано на языке ассемблера.

Языки высокого уровня были разработаны для того, чтобы освободить программиста от учета технических особенностей конкретных компьютеров, их архитектуры.

Языки высокого уровня делятся на:

1) процедурные (алгоритмические) (*Basic*, *Pascal*, *C* и др.), которые предназначены для однозначного описания алгоритмов; для решения задачи процедурные языки требуют в той или иной форме явно записать процедуру ее решения;

2) 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;

3) 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.

Один и тот же язык программирования может быть реализован по-разному даже на одном и том же компьютере (пример: *GW Basic* и *QBasic*). В синтаксисе различных реализаций допускается некоторые второстепенные отличия.

Вопрос 2. Основные этапы разработки программ

Чаще всего при написании реальной прикладной программы требуется не только программирование. Сначала необходимо получить информацию для разработки программы, на основе которой выбирается подходящий метод. Кроме того, когда программа будет написана, введена и отлажена, обычно проводится анализ полученных числовых результатов и делаются выводы относительно исходной задачи.

В итоге можно выделить следующие основные этапы разработки программ.

1. Постановка задачи – выяснение наиболее существенных ее параметров и закономерностей, влияющих на изучаемую ситуацию; формулировка целей решения и определение необходимых для этого данных.

2. Математическая формализация – запись основных закономерностей в математической формуле: в виде уравнений, соотношений, связей, условий.

3. Выбор компьютерного метода – обычно данный этап требуется, поскольку получившаяся математическая задача нуждается в преобразовании в эквивалентную форму, пригодную для обработки на компьютере, – например, дифференциальные уравнения приводятся к алгебраическим формулам, а для статистического моделирования требуется реализация датчика случайных чисел.

4. Построение алгоритма.

5. Составление программы на языке программирования.

6. Отладка и тестирование программы – установление синтаксических и логических ошибок, анализ полученных результатов на предмет согласованности с имеющимися закономерностями и ранее надежно установленными результатами.

7. Проведение расчетов и анализ полученных результатов.

Вопрос 3. Технология программирования

Программирование – это совокупность процессов, связанных с разработкой программ и их реализацией. Имеющиеся программные и технические средства постоянно совершенствуются, в результате чего появляются новые методы, методологии и технологии, которые в дальнейшем служат основой более современных средств разработки программного обеспечения. Возникает необходимость отслеживать процесс создания новых технологий и определять их основные тенденции развития, проводя параллель между этими технологиями, уровнем развития программирования и, как следствие, особенностями программных и аппаратных средств.

Технологией программирования называют совокупность методов и средств, используемых в процессе разработки программного обеспечения. Как любая другая технология, технология программирования представляет собой набор инструкций, включающих:

- указания последовательности выполнения технологических операций;
- перечисление условий, при которых выполняется та или иная операция;
- описания самих операций, причем для каждой операции определены исходные данные, результаты, а также правила, нормативы, стандарты, критерии и методы оценки и т.п.

Технология программирования определяет набор операций и их последовательность выполнения, а также на каждом этапе разработки модель проектируемой системы.

Технологии могут быть использованы для решения отдельных задач или определять выполнение этапа разработки, кроме того, технологии могут охватывать несколько этапов или весь процесс разработки.

На выбранном программистом языке программирования пишется исходный код, который затем преобразуется транслятором в объектный модуль, хранимый впоследствии на машинном носителе. **Транслятором** называется программа, осуществляющая перевод программы с языка программирования в машинный код – язык конкретной вычислительной машины. **Объектный модуль** – последовательность команд машинного кода, полученная в результате работы компилятора. Объединение модулей объектного кода в единую исполняемую программу выполняет компоновщик.

В теории программирования существуют две взаимосвязанные составляющие процесса решения задачи: собственно данные и инструкции по их обработке, т.е. алгоритм.

Каждый алгоритм – это правила, описывающие процесс преобразования исходных данных в необходимый результат. Чаще всего алгоритмы предполагают обработку некоторых величин. **Величина** – это элемент данных с точки зрения их смыслового (семантического) содержания или обработки. При разработке алгоритма данные можно разбить по смыслу на входные – аргументы, выходные – результаты, и промежуточные. **Исходные** (входные) – это данные, известные перед выполнением задачи из условия. **Выходные данные** – результат решения задачи. Переменные, которые не являются ни аргументом, ни результатом алгоритма, а используются только для обозначения вычисляемого промежуточного значения, называются **промежуточными**. Кроме того требуется указать имена и типы данных – целый, вещественный, логический и символьный, либо структурированный, базирующийся на одном из названных.

Основными понятиями, используемыми при программировании, обычно

являются следующие:

1. **Имена** (идентификаторы) – употребляются для обозначения объектов программы (переменных, массивов, функций и др.).

2. Операции и типы операций:

- арифметические операции: +, –, *, / и др.;
- логические операции: и, или, не;
- операции отношения:<, >, <=, >=, =, <>;
- операция присоединения или конкатенации символьных значений друг с другом с образованием одной длинной строки; изображается знаком «+».

3. Данные – величины, обрабатываемые программой.

Данные подразделяют на переменные и константы.

Константы – это данные, которые зафиксированы в тексте программы и не изменяются в процессе ее выполнения, например, числовые – 7.5,12;логические – да (истина), нет (ложь); символьные (содержат ровно один символ) – "A","+"; литерные (содержат произвольное количество символов) – "d23", "Мир".

Переменные обозначаются именами и могут изменять свои значения в ходе выполнения программы.

В конкретном языке программирования каждой величине соответствует своя переменная. Помимо вида, каждая величина в алгоритме имеет свой тип**.** Тип величины показывает, какие значения может принимать величина и какие операции можно с ней выполнять.

Основные стандартные простые типы данных, использующиеся в алгоритмических языках:

- числовые типы данных делятся на вещественные и целые, которые могут рассматриваться как содержащие знак, так и беззнаковые. Кроме того, конкретные реализации языков программирования чаще всего содержат несколько разновидностей целых и вещественных данных, что связанно с различным объемом памяти, выделяемым для них;
- символьные типы данных служат для хранения и организации обработки текстовой информации;
- логические (булевские) типы данных – логические переменные имеют всего два значения – «ложь» и «истина»;
- вариантный тип данных, который появился в современных реализациях языков программирования и может принимать любые (числовые, символьные и т.д.) значения, причем тип текущего значения также хранятся в самой переменной, и может быть запрошен программистом. Подобный тип данных лежит в основе языка *VBA* (*VisualBasicforApplications*); на его

базе реализуется общий формат ячеек электронной таблицы *Excel*.

Языки программирования позволяют сконструировать и сложные типы данных, причем для этого можно объединять как несколько простых, так и другие сложные данные. Самым известным сложным типом является массив. Массивы – последовательности однотипных элементов, число которых фиксировано и которым присвоено одно имя. Положение элемента в массиве однозначноопределяется его индексами (одним, в случае одномерного массива, или не- сколькими, если массив многомерный). Иногда массивы называют таблицами.

На простой стандартный тип можно непосредственно ссылаться, а для сложного типа требуется предварительно описывать его структуру. Во многих языках существует возможность определения собственного типа данных.

4. **Выражения** предназначаются для выполнения необходимых вычислений, состоят из констант, переменных, функций, объединенных знаками операций. Различают выражения арифметические, логические и строковые.

Арифметическое выражение состоит из арифметических констант, переменных и функций, соединенных знаками арифметических операций и скобками. Например, (y–*Q*/5 + sin(*x*)). Частным случаем арифметического выражения является переменная, функция или константа.

Логическое выражение состоит из арифметических и логических констант, переменных и функций, соединенных знаками сравнения, логических операций и круглыми скобками. Логическое выражение может принимать только два значения – «истина» или «ложь» (да или нет). Например, $x + y<5$.

Строковые выражения, значениями которых являются тексты. В строковые выражения могут входить литерные и строковые константы, литерные и строковые переменные, литерные функции, разделенные знаками операции сцепки. Например, A = "русский", а B = " язык", тогда значение выражения A + B = "русский язык".

5. **Оператор** – это команда, записанная на специальном языке и определяющая некоторое действие по обработке данных или описание структуры программы. 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.).

6. **Подпрограмма** – это специальным образом оформленный алгоритм, который может многократно использоваться при решении более общей задачи. Использование подпрограмм дает возможность разрабатывать программу по частям, проверяя на правильность каждую из частей. В программе подпрограмм

может быть несколько.

При решении новых задач часто используется метод аналогий, т.е. ранее разработанный алгоритм целиком используется в составе других алгоритмов – вспомогательный. Применение вспомогательных алгоритмов позволяет разбить задачу на основную и вспомогательную части, т.е. структурировать ее.

В основной части производится простейшая обработка информации, организуется обращение к разным подпрограммам. Вспомогательный алгоритм тоже может вызывать другие вспомогательные. При использовании вспомогательных алгоритмов необходимо учитывать способ передачи значений исходных данных для них и получения результата от них. Результаты вспомогательного алгоритма – это также переменные, где содержатся результаты решения этих подзадач, также результатом может быть конкретное действие, которое совершает компьютер под действием подпрограммы.

При решении задач целесообразно проанализировать условие, записать решение в крупных блоках, детализировать каждый из блоков и т.д., продолжать до тех пор, пока каждый из блоков не будет реализован с помощью операторов языка.

Существует большое количество способов организации обработки данных. Особо можно выделить **итерационный** и **рекурсивный** методы.

Каждое вычисление, приближающее к решению поставленной задачи, причем результат обработки всегда служит начальным состоянием для нового просмотра, в компьютерной литературе называют **итерацией**. Т.е. итерация – это повторение пошагового процесса, когда результат предыдущего шага (шагов) используется для получения результата следующего шага.

Возможны также подпрограммы, которые вызывают сами себя. Они называются рекурсивными. **Рекурсия** – это способ решения задачи, при котором производится последовательное ее сведение к аналогичной, но более простой. К рекурсии в принципе можно свести любой циклический алгоритм, но не наоборот.

Существует несколько принципиальных подходов к программированию.

Структурный подход к программированию предполагает на первом этапе формирование стратегических параметров программного обеспечения и определение общего алгоритма работы программы, а затем его разбиение (декомпозиция) на части до уровня небольших подпрограмм – проектирование осуществляется «сверху вниз». Одновременно с этим вводятся ограничения на конструкции алгоритмов, предлагаются формальные модели их описания, а также специальный метод проектирования алгоритмов – **метод пошаговой детализации.** Далее подпрограммы объединяют в единую программу.

С появлением других принципов декомпозиции (объектного, логического

и т.д.) данный способ получил название процедурной декомпозиции. Кроме того, при использовании структурного подхода программирования при составлении программ используются только базовые алгоритмические структуры, и существует запрет на использование оператора *goto*.

Языки, поддерживающие подход структурного программирования, получили название **процедурных языков программирования**. Среди наиболее известных языков этой группы стоит назвать *PL*/1, *ALGOL*-68, *Pascal*, *C*.

Модульный подход к программированию предполагает деление программы на модули, которые при компиляции обрабатывают отдельные объектные файлы, которые затем собираются в единое целое и формируют исполняемую программу. В этом случае существует возможность объединять фрагменты программы по функциональному признаку.

Связи между модулями выполняются через специальный интерфейс, но доступ к внутренней структуре модуля и некоторым «внутренним» локальным переменным запрещен. Эту технологию поддерживают современные версии языков *Pascal* и *C* (*C++*), языка Ада и *Modula*.

Подход **объектно-ориентированного программирования** определяется как технология создания сложного программного обеспечения, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного типа (класса), а классы образуют иерархию с наследованием свойств. Взаимодействие программных объектов в такой системе осуществляется путем передачи сообщений.

Основным достоинством объектно-ориентированного программирования является более полная детализация данных и интегрирование их с подпрограммами обработки, что позволяет вести практически независимую разработку отдельных частей (объектов) программы. Кроме этого, данный подход использует новые способы организации программ, основанные на механизмах наследования, полиморфизма, композиций, наполнения, что позволяет конструировать сложные объекты из сравнительно простых. В результате существенно увеличивается показатель повторного создания библиотек классов для различных применений.

Среды, поддерживающие **визуальное программирование**, – *Delphi*, *C++ Builder*, *VisualC++* и т.д. При использовании визуальной среды у программиста появляется возможность проектировать интерфейсы будущего продукта, с применением визуальных средств добавления и настройки специальных библиотечных компонентов. Результатом визуального проектирования является заготовка будущей программы, в которую уже внесены соответствующие коды.

Основными понятиями, используемыми в объектно-ориентированном программировании являются:

1. **Объект**– это совокупность переменных состояния и связанных с ними

методов (операций), определяющих поведение объекта и способы его взаимодействия с окружающим миром.

2. **Инкапсуляция** – это механизм, который объединяет данные и методы, манипулирующие этими данными, и защищает и то и другое от внешнего вмешательства или неправильного использования. Применение инкапсуляции позволяет защитить данные, принадлежащие объекту, от возможных ошибок, которые могут возникнуть при прямом доступе к этим данным. Кроме того, применение этого принципа очень часто помогает локализовать возможные ошибки в коде программы. Однако применение инкапсуляции ведет к снижению эффективности доступа к элементам объекта, что обусловлено необходимостью вызова методов для изменения внутренних элементов (переменных) объекта.

3. **Наследование** – это процесс, посредством которого один объект может наследовать свойства другого объекта и добавлять к ним черты, характерные только для него. В итоге создается иерархия объектных типов, где поля данных и методов «предков» автоматически являются и полями данных и методов «потомков», т.е. не надо каждый раз заново описывать новый объект, а можно указать «родителя» (базовый класс) и описать отличительные особенности нового класса. В результате новый объект будет обладать всеми свойствами родительского класса плюс своими собственными отличительными особенностями.

4. **Полиморфизм** подразумевает такое определение методов в иерархии типов, при котором метод с одним именем может применяться к различным родственным объектам. Преимуществом полиморфизма является то, что он помогает снижать сложность программ, разрешая использование одного интерфейса для единого класса действий. Выбор конкретного действия, в зависимости от ситуации, возлагается на компилятор.

Компонентный подход – технология программирования, которая предполагает построение программного обеспечения из отдельных компонентов – физически отдельно существующих частей программного обеспечения, которые взаимодействуют между собой через стандартизованные двоичные интерфейсы. В отличие от обычных объектов объекты-компоненты можно собрать в динамически вызываемые библиотеки или исполняемые файлы, распространять в двоичном виде и использовать в любом языке программирования, поддерживающем соответствующую технологию.

Компонентный подход лежит в основе технологий, разработанных на базе *COM* (*ComponentObjectModel* – компонентная модель объектов), и технологии создания распределенных приложений *CORBA* (*CommonObjectRequestBrokerArchitecture* – общая архитектура с посредником обработки запросов объектов). Эти технологии используют сходные принципы и различаются лишь особенностями их реализации.

Вопрос 4. Использование языка программирования *Visual Basic*

Visual Basic (*VB*) принадлежит к разряду новых инструментов программирования – к средам визуальной разработки программ. Он сочетает производительность и относительную простоту программирования с набором визуальных элементов управления, при помощи которых строятся программы.

Visual Basic является средой программирования в ОС *Windows* и используется для написания *Windows*-программ. Разработка программного интерфейса осуществляется на принципах объектно-ориентированного подхода, реализованного в *VB* применительно к приложениям, выполняемым под управлением ОС *Windows*.

Характерной особенностью *VB* является возможность создания и использования самостоятельных компонентов – объектов. Одним из типов объектов являются элементы управления: окна, кнопки, меню и т.д., с помощью которых можно дополнять программы новыми функциями, не вникая при этом в суть их работы.

Для каждого объекта существует ряд возможных событий, например, нажатие кнопки мыши или выбор пункта меню. События проявляются в действиях, реакции программы и могут быть следствием свойств объекта или результатом деятельности программиста. Т.е. для каждого объекта события существует возможность задания процедуры *Visual Basic*. Первоначально каждая такая процедура является пустой и не отвечает действием на произошедшее событие, но программист может заполнить кодом процедуру для любого события, определив, таким образом, его возможности.

Таким образом, *VB* предоставляет пользователю объектно-ориентированную среду программирования, основанную на событиях, где процесс разработки программы сводиться к выбору набора объектов и их свойств, заданию событий и процедур их обработки, которые в совокупности обеспечивают решение поставленной задачи.

При первом запуске *Visual Basic* открывается окно, в котором указан тип программного проекта (приложения *VB*), установленный по умолчанию – **Standard EXE** (рис. 45). Все функции данного окна заложены в меню **Файл**.

Меню содержит три пункта:

- **New** – начало создания нового проекта;
- **Existing** – выбор приложения из существующих проектов;
- **Recent** – список нескольких последних проектов.

Рис. 45. Стартовое окно *Visual Basic*

Новый проект открывается в среде программирования *IDE* (интегрированная среда разработки) *Visual Basic* вместе с некоторыми другими окнами и инструментами (рис. 46). Данное окно содержит все элементы, которыми обладает любое приложение *MSWindows* – строки заголовка, меню и инструментов.

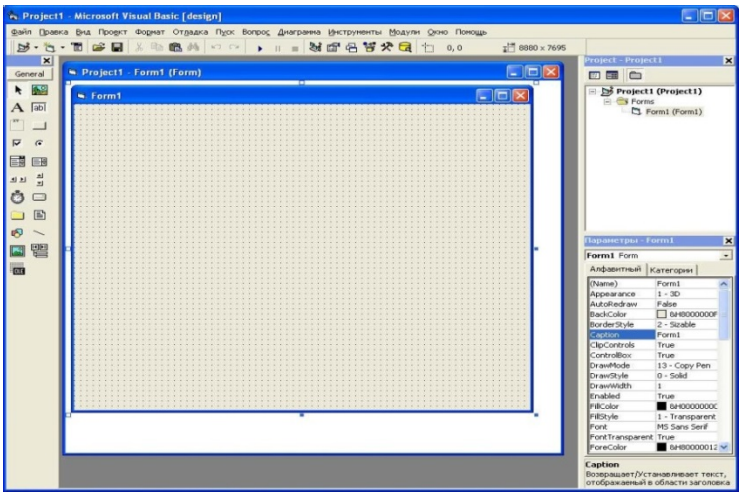


Рис. 46. Окно создания пользовательского интерфейса нового проекта в*Visual Basic*

При сохранении проекта необходимо, прежде всего, сохранить в отдельных файлах все его компоненты – формы, модули программного кода и классов и т.д., а затем и сам проект в соответствующем файле. Лучше всего для каждого нового проекта создавать отдельную папку, что позволит хранить все модули проекта в одном месте. Для сохранения проектов необходимо воспользоваться командами меню **Файл**.

В окне располагаются:

- окно инструментов с пустой строкой заголовка (**Toolbox**);
- окно содержания проекта (**ProjectContainer**);
- окно формы, находящееся внутри окна содержания проекта (**Form1**);
- окно проекта (**Project**);

- окно свойств (**Properties**);
- окно макета формы (**FormLayout**);
- окно непосредственного выполнения (**Immediate**).

Окно **Form1** представляет собой экранную форму со стандартной сеткой, в которой программисты располагают различные элементы управления, представленные на панели инструментов в виде набора кнопок, обеспечивающих выполнение определенных команд:

- надпись, для отображения пояснений (Label) A;
- поле для ввода текста (TextBox) abl;
- командная кнопка (CommandButton);
- флажок (Check Box);
- переключатель (OptionButton);
- поле со списком (ComboBox);
- список (ListBox);
- полосы прокрутки (ScrollBar) вертикальная и горизонтальная;
- таймер(Timer);
- контейнер OLE (OLE Container) др.

Элемент управления можно добавить в форму перетащив его с панели инструментов мышью, таким образом, сводя к минимуму непосредственное написание программного кода. Размер элемента управления можно настроить, используя маркеры изменения размеров. Для уничтожения элементаего необ- ходимо выделить и нажать клавишу *Delete* либо воспользоваться контекстным меню.

После того как элемент управления помещен в форму, можно изменить его свойства – имя, надписи, размер и положение и т.д. (табл. 10), и функции.

Каждый элемент управления должен иметь уникальное имя, которое хра- нится в свойстве **Name**., которое затем будет использоваться в программе для идентификации. Для изменения свойств выбранного элемента используется ок- но свойств с названием **Properties**, за которым следует имя элемента управле- ния. В соответствующих полях данного окна и производятся изменения свойств.

Вычислительная логика может быть встроена в приложение только с по- мощью программного кода, который точно определяет, что именно должна де- лать программа на каждом этапе работы. Программа управляется элементами управления, поэтому программный код будет связан с информацией, посту- пающей от этих элементов.

Таблица 10

Основные общие свойства элементов управления

Название свойства	Описание свойства
Name	Имя элемента управления
AutoSize	Определяет возможность автоматического изменения размера при заполнение (допустимые значения: *true* – режим автоматического изменения размеров, чтобы полностью помещался текст, присвоенный свойству Caption, и *false* – размер не меняется автоматически).
BackColor, ForeColor и BorderColor	Устанавливают цвет заднего и переднего плана элемента управления, также его границы.
BackStyle	Устанавливает тип заднего фона.
BorderStyle	Устанавливает тип границы.
ControlTipText	Устанавливает текст в окне всплывающей подсказки, связанной с элементом управления.
Caption	Надпись, отображаемая при элементе управления
Enabled	Определяет возможность доступа к элементу (допустимые значения: *true* – пользователь вручную может управлять элементом управления, и *false* – ручное управление отсутствует).
Height и Width	Устанавливают геометрические размеры объекта (высоту и ширину).
Left и Top	Устанавливают координаты верхнего левого угла элемента управления, определяющие его местоположение в форме.
Picture	Внедряет картинку на элемент управления.
Tag	Определяет идентификацию внешних данных.
Visible	Определяет видимость элемента управления во время выполнения программы (допустимые значения: *true* – элемент управления отображается, и *false* – не отображается).

Для работы с программным кодом элемента нужно вызвать окно **Код** (Cod) двойным щелчком мыши на элементе.

Блок кода, связанный с частным объектом интерфейса, называется процедурой события *Visual Basic.* Тело процедуры заключено между операторами, указывающими на начало и конец подпрограммы.

Private Sub ИмяЭлемента
Операторы *Visual Basic*
End Sub

Операторы тела и процедуры выполняются каждый раз, когда пользова- тель активизирует элемент интерфейса, ассоциированный с процедурой. *Я*зык *Visual Basic* содержит развитые средства контекстной оперативной помощи пользователю при наборе программного кода. После завершения работы над программой ее нужно сохранить.

Запуск программы может быть осуществлен из меню **Run** или щелчком мыши по кнопке **Start** на панели инструментов.

Visual Basic является языком программирования, и как любой другой язык имеет свой алфавит, используемый для написания операторов или предложе- ний. Алфавит *VB* включает:

1) латинские буквы: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z;

2) арабские цифры: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0;

3) символы: !#$%&л*()+-/\<>=?@_|:'.,“ ” пробел;

4) другие символы, включая русские, можно использовать только в строках символов.

Данные в *VB* характеризуются типом и организацией. В табл. 18 приводят- ся характеристики используемых типов данных.

Для хранения постоянных величин *Visual Basic* позволяет объявить константы, т.е. выделить участки памяти, содержимое которых не меняется. Объявление констант осуществляется оператором:

Public | PrivateConstимя[Astype] = выражение

где Public– константу можно использовать в любых процедурах и функциях;

Private– константу можно использовать только внутри модуля, в котором она определена.

Имя констант принято записывать прописными буквами. Например, Const V=3.142 или Const MY_NAME = ″Света″.

Тип константы можно не объявлять, но удобнее явно указать тип специальными символами в операторах объявления констант. Используемые символы показаны в табл. 11.

Например: ConstONE&= 1

В *Visual Basic* имеется большое число встроенных констант, значения которых определены заранее и используются без предварительного определения. Например,vbOKCancel = 1 – аргумент функции MsgBox для вывода в диалоговом окне командных кнопок *OK* и *Cancel*. В файле *VB*constant.txt содержатсячасто используемые константы, которые могут быть скопированы в программу.

Переменные в программе могут хранить различные значения и их перед использованием нужно объявлять. При объявлении переменной надо задать имя переменной и указать ее тип.

Имена в*Visual Basic* используются для обозначения объектов в программе. Правила образования имен:

- первым символом имени должна быть латинская буква;

- имя может включать только латинские буквы, цифры и знак подчеркивания;

Таблица 11

Характеристики типов данных

Тип данных	Описание	Диапазон значений	Занимаемая память
Byte	Целые числа	от 0 до 255	1 байт
Integer	Целые числа одинарной точности	от – 32768 до + 32767	2 байта
Long	Целые числа двойной точности	от – 2 147 483 648 до +2 147 483 647	4 байта
Currency	Число с фиксированной десятичной точкой	$\pm (9 \cdot 10^{14})$	8 байтов
Single	Вещественные числа одинарной точности	от $\pm 1.4 \cdot 10^{-45}$ до $\pm 3.4 \cdot 10^{38}$	4 байта
Double	Вещественные числа двойной точности	от $\pm 4.94 \cdot 10^{-324}$ до $\pm 1.79 \cdot 10^{308}$	8 байтов
Date	Дата	от 1 / January /100 до 31 / December / 9999	8 байтов
Boolean	Логический	true или false	2 байта
Object	Ссылка на объект OLE		4 байта
Variant	Произвольный тип	способен принимать любые значения	зависит от значения

Таблица 12

Характеристики типов данных

Символ объявления типа	Тип данных
%	integer
&	long
1	single
#	double
@	currency
$	string

- имя может содержать не более 40 символов;ключевые или зарезервированные слова не могут использоваться в качестве имен (список ключевых слов содержится в справочной системе *Visual Basic* в разделе *Reserved word*).

Примеры имен: strMyName, i, strInputValue, N и т.д.

Существуют соглашения по стилю имен, которых желательно придерживаться:

- идентификатор должен понятным образом отражать назначение переменной, что способствует пониманию программы;
- лучше использовать имена из строчных букв, а в случае составного имени слова отделять друг от друга знаком подчеркивания или начинать новое слово с прописной буквы (например, Фамилия Студента, intNumber1);
- имена из прописных букв используются для определения констант (MY_NAME);
- название идентификатора нужно начинать со специального знака (приставки или префикса), который указывает на тип данных, связанный с этим идентификатором (inrNalog).

Объявление переменной может быть явным или неявным. Для явного определения переменных можно использовать следующий синтаксис:

- **Dim ИмяПеременной1 As ТипПеременной1 [, ИмяПеременной2 As ТипПеременной2]**;
- **Private ИмяПеременной1 As ТипПеременной1 [, ИмяПеременной2 As ТипПеременной2]**;
- **Public ИмяПеременной1 As ТипПеременной1 [, ИмяПеременной2 As ТипПеременной2]**;
- **Static ИмяПеременной1 As ТипПеременной1 [, ИмяПеременной2 As ТипПеременной2]**.

где Dim (Размер) – ключевое слово, показывающее объявление переменной и резервирующее область памяти для ее хранения.

Private (Частный)и Public (Общий)– ключевые слова, определяющие область видимости переменной.

Static (Статический) – ключевое слово, которое определяет, сохраняет ли переменная свое значение при завершении подпрограммы и выходе из нее.

Например: Private nNumValAsInteger, Dim sFirstName As String.

Так же при явном определении переменной можно использовать суффиксы, добавляя в конец ее имени специальный символ описания типа (табл. 17):

- **Dim ИмяПеременнойСуффикс**;
- **Private ИмяПеременнойСуффикс**;
- **Public ИмяПеременнойСуффикс**;
- **Static ИмяПеременнойСуффикс**.

Например: Dim strInputMsg$, Static intNumVal%.

Неявное объявление переменных осуществляется при использовании оператора DefТип_Данных, который устанавливает тип данных для переменных,

параметров процедур и тип возвращаемого значения для процедур типа Functionи Property Get, имена которых начинаются с определенного символа:

DefТип_Данных ДиапазонБукв [,ДиапазонБукв].

Например: DefIntA-K (переменные, имена которых начинаются с символов A-K, будут иметь тип Integer).

Visual Basic допускает использование имен без объявления их типа (в этом случае автоматически определяется тип, требующий для размещения минимальной памяти).

Существует понятие области действия (scope) данных, определяющее возможность доступа к тем или данным (например, к переменной, константе, массиву и т.д.) в отдельных процедурах одной формы или в процедурах, относящихся к разным формам одной программы.

Если оператор объявления какой-либо переменной находится внутри процедуры обработки события, то использованиеэтой переменной возможно только в рамках данной процедуры. Такая переменная называется локальной (local). Локальные данные определены только в момент выполнения процедуры или функции и память для них выделяется только на момент выполнения.

Для того чтобы одна и та же переменная могла использоваться в разных процедурах одной формы, оператор объявления переменных должен быть помещен в раздел общих объявлений (general), доступ к которому открывается щелчком мыши по элементу **general** раскрывающегося списка окна **Object** формы. Объявленная таким образом переменная имеет статус действующей на уровне модуля (modul-level variable) и может использоваться в любой процедуре данной формы. Для того чтобы одна и та же переменная могла использоваться в процедурах разных форм одной программы она должна быть объявлена как глобальная переменная (global variable), при этом используется ключевое слово Global вместо Dim. Глобальные данные сохраняются, пока выполняется программа, и они доступны для всех входящих в программу процедур и функции. При запуске программы *Visual Basic* отводит для глобальных данных необходимую память.

Например: GlobalName, YourNameAs String,NAs Integer.

Операторы объявления глобальных переменных помещаются в модулях кода (code modules), которые могут использоваться во всей программе. Программы, состоящие из нескольких форм и соответственно из нескольких файлов, размещаются в файлах модулей кода с расширением. Такие файлы создаются при выборе в меню **Вставить** (Insert) команды **Создать модуль** (Module) или при щелчке мыши на одноименной кнопке панели инструментов.

Данные различных типов можно сгруппировать по какому-либо признаку в удобную для использования одну структуру – запись (records).

Оператор объявления пользовательского типа данных (записи) помещается в модуль и имеет вид:

TypeИмяЗаписи
ИмяЭлементаЗаписи1AsТипЭлементаЗаписи1
ИмяЭлементаЗаписи2 As ТипЭлементаЗаписи2
. . .
ИмяЭлементаЗаписиNAsТипЭлементаЗаписиN
EndType

где ИмяЗаписи и ИмяЭлементаЗаписи – образуются в соответствии с общими правилами образования имен в *VB*;

Например:

```
Type  Student
strФамилия  As  String
НомерГруппыAsInteger
EndType
```

Для обращения к конкретному элементу структуры определенного объекта используется составная запись **ИмяПеременной.Имя СтруктурногоЭлемента = выражение**.

Например: udtStudent.strФамилия = "Степанов"

В *Visual Basic*кроме простых переменных могут быть использованы переменные структурного типа. Самым распространенным сложным типом является **массив,** соединяющий в себе набор данных одного типа.

Массив – упорядоченный набор однотипных данных, обозначенный одним именем. Массив может строится из однотипных переменных, однородных пользовательских типов данных (одинаковых записей), однотипных элементов пользовательских типов данных.

Для сложного типа в обязательном порядке требуется предварительно описать его состав.

Обращение к отдельным элементам массива производится путем указания общего имени массива и определенной информации, характеризующей позицию требуемого элемента внутри массива. Последняя имеет смысл некоторой координаты (или, может быть, нескольких координат) и называется **индексом** (индексами). Отсюда видно, что массивы естественным образом делятся по количеству индексов на **одномерные** (с одним индексом), **двумерные** и, что допускается в большинстве языков, многомерные массивы (с большим числом индексов).

Массив объявляется уже рассмотренным оператором

Dim |Public | PrivateИмяМассива (нижняя граница , | To верхняя граница) AsТипДанныхМассива

Где Dim, Public, Private – ключевые слова;

ИмяМассива – образуется в соответствии с общими правилами образования имен в *VB*;

To – ключевое слово, разделяющее нижнююи верхнюю границу изменения индекса массива;

нижняя граница – определяет минимальное значение индекса массива (целого типа);

верхняя граница – определяет максимальное значение индекса массива (целого типа).

Количество повторений [нижняя граница To верхняя граница] определяет размерность массива (количество индексов, используемое для определения элементов массива). Максимальное число индексов равно 60.

Например: DimA(10)AsIntegerилиDimA(8, 0 To 3) As Double.

В *Visual Basic* имеется возможность переопределять количество элементов массива в момент выполнения программы, т.е. динамически определять размер массива. Это позволяет эффективно использовать дефицитную оперативную память при создании программ. Для этого в программе используется оператор ReDim:

ReDim [Preserve] ИмяМассива (нижняя граница , | To верхняя граница) [As ТипДанныхМассива]

Preserve сохраняет данные существующего массива при выполнении оператора (для многомерных массивов можно применять только для правого индекса).

Элементы объявленного массива могут использоваться в операторах программы в виде**ИмяМассива (значение индекса [, значение индекса] ...)**.

Значение индекса задается арифметическим выражением, нецелочисленные значения выражения округляются до ближайшего целого, значения выражения должны находиться в пределах возможных значении индекса, задаваемых операторами определения или переопределения массива.

Если массив объявлен оператором DimA(10,20)As Double,то для обращения к его элементам в операторах программы можно использоватьA(0,0), A(5,j+ 1) или A(2.5, 9.8).

В списке свойств элементов управления имеется свойство Index, значение

которого определяет индекс данного элемента в массиве. Индексы присваиваются автоматически последовательно при создании на форме нового одинакового элемента управления (первому элементу – 0, второму – 1 и т. д.) Эта последовательность может быть изменена разработчиком формы, но только на этапе создания формы.

Выражения используются для операций над данными. В зависимости от данных и используемых операций выражения можно разделить на арифметические, логические и символьные.

Арифметическое выражение использует следующие знаки операций: + (сложение); – (вычитание);* (умножение); ^ (возведение в степень);/ (деление с плавающей точкой);\ (целочисленное деление); Mod (вычисление остатка от деления).

Операндами выражения могут быть константа; переменная; элементмассива; обращение к стандартной функции; обращение к процедуре – функции; арифметическое выражение в скобках.

Логические выражения используются в математической логике и их также называют булевыми выражениями.

Используются следующие знаки логических операций:

- Not (логическое отрицаниеНЕ);
- And (логическое умножение И);
- Or (логическое сложение ИЛИ);
- Xor (исключительное ИЛИ);
- Eqv (логическая эквивалентность);
- Imp (логическая импликация).

Выражения отношения состоят из двух арифметических или символьных выражений, объединенных знаками операций отношения:

- > (больше);
- <(меньше);
- >= (больше или равно);
- <= (меньше или равно);
- = (равно);
- <>(не равно).

Логические операции объединяют логические величины, которые могут принимать два значения: *true* (истина) или *false* (ложь). Результат логической операции также принимает одно из двух значений: *true* (истина) или *false* (ложь).

Приоритет выполнения операций (в порядке убывания приоритета): Not, And, Or, Xor, Eqv, Imp.

Например: True And Not False Or False.

Операндами логического выражения являются: логические константы; логические переменные; обращения к функциям, возвращающим логические значения; выражения отношения; заключенные в скобки логические выражения.

Двойные неравенства для правильного их вычисления необходимо записывать с использованием знаков логических операций. Например,25< AAndA< 55.

В *Visual Basic* определена одна операция с символьными данными – сцеп- ление, позволяющая объединять несколько строк в одну. Знак операции «+» или «&».

Операндами символьного выражения могут бытьсимвольная константа; символьная переменная; элемент символьного массива (string);обращение к процедуре-функции, возвращающей символьное значение;обращение к стандартной функции, возвращающей символьное значение.

Например: сцепление Name$ =″оборотный″ и SurName$ = ″капитал″ можно записать Narne$ + SurName$ = ″оборотныйкапитал″.

В *VB* имеется широкий набор встроенных (стандартных) функций, облегчающий написание программ (табл. 13). Обращение к встроенной функции, возвращающей значение того или иного типа, должно соответствовать выражению, в котором к ней обращаются. Например, и арифметическом выражении можно обращаться к функциям, возвращающим значения арифметических типов, в символьном – символьного типа. Для обращения к некоторым встроенным функциям нужно задавать значение аргумента.

Таблица 13

Примеры функций *Visual Basic*

Имя функции	Возвращаемое значение
Математические функции	
Abs	абсолютное значение (модуль) числа
Atn	арктангенс от значения параметра, заданного в радианах
Cos	косинус указанного в радианах угла
Exp	экспонента числа e^x
Int	результат округления выражения с плавающей точкой до целой части
Log	натуральный логарифм числа
Rnd	случайное число одинарной точности в интервале от 0 до 1
Sgn	знак числа
Sin	синус указанного в радианах угла
Sqr	квадратный корень числа
Tan	тангенс от значения параметра, заданного в радианах
Функции обработки строк	
Asc	ASCII-код символа
Chr	символ по ASCII-коду
Instr	номер позиции первого символа подстроки в строке

Lcase	преобразовывает строку в нижний регистр
Len	определяет длину строки
Spase	создает строку пробелов
Str	преобразовывает число в строку
StrComp	сравнивает две строки
String	создает строку символов
Ucase	преобразовывает строку в верхний регистр
Val	преобразовывает строку в число
Функции даты и времени	
Date	устанавливает и возвращает текущую дату
DateSerial	преобразовывает в дату три целых числа (день, месяц, год)
Day	число от 1 до 31, соответствующее числу месяца
Month	число от 1 до 12, соответствующее месяцу
Time	устанавливает и возвращает текущее время
Year	число, соответствующее году
Финансово-математические функции	
DDB	амортизация фондов в течении заданного интервала времени
FV	накопленная стоимость при известном размере регулярного взноса и постоянной процентной ставке
IRR	внутренняя норма доходности при известной последовательности выплат и поступлений
MIRR	модифицированная внутренняя норма доходности
NPV	чистая приведенная стоимость инвестиционного проекта при известном размере выплат и поступлений п при постоянной дисконтной ставке
SLN	величина амортизации фондов линейным методом
SYD	величина годовой амортизации фондов за определенный период

Программа на *Visual Basic* состоит из процедур, которые в свою очередь состоят из операторов. **Оператором** является синтаксически полное описание конкретной команды, которая выражает одно действие или определение.

Одному оператору соответствует одна строка программы. Однако можно использовать разделительный знак двоеточие (:), чтобы поместить больше чем один оператор в строке программы.

Строки программы могут быть обозначены метками (Linelabel) или номерами (Linenumber).

Метка (Label) обозначает следующую строку программы. Метка может включать не более 40 символов (первый обязательно буква) и заканчивается двоеточием(:), не может быть ключевым словом. Метка может начинаться в любой позиции строки, если ей не предшествует никакой символ.

Номер строки (Lilienumber) обозначает следующую строку программы. Номер строки может включать не более 40 десятичных цифр ине заканчивается

двоеточием. Номер строки может начинаться в любой позиции строки, если ему не предшествует никакой символ. В рамках одной процедуры номера строк не могут повторяться.

Оператор присваивания имеет следующий вид:

ИмяПеременной = выражение.

Переменной или элементу массива в левой части оператора присваивается значение вычисленного выражения правой части.

Например: intI = 6.

Если в левой части оператора используется переменная или элемент массива символьного типа, то выражение в правой части должно быть тоже символьное. Если в левой и правой частях оператора используются арифметические данные, но разных типов, то тип правой части преобразуется к типу левой части. Результатом присвоения значения вещественной константы 3.5 переменной целого типа (I = 3.5) будет 3.

Для описания алгоритма обработки информации и реализующейего программы, содержащих проверку какого-либо условия, от которых зависит последующее действие, используется условный оператор, который имеет вид:

If логическое выражение Then выражение– безальтернативная форма;

If логическое выражение Then выражение1Else выражение2 – альтернативная форма.

В операторе выражение – это один или несколько операторов (разделеныдвоеточием).

В безальтернативной формепри значении логического выражения *true* сначала выполняется выражение, следующее за словом Then, а затем следующий оператор. Если логическое выражение принимает значение *false*, то выражение не выполняется.

Например: If A > 10 Then A = A + I

В альтернативной форме условный оператор обеспечивает выполнение выражения 1 или выражения 2 в зависимости от значения логического выражения, которое принимает значение *true* или *false*.

Например: If C> =10 Then A = 5 Else A = 10.

Условный оператор может иметь блочную конструкцию, если в случае истинности и ложности логического выражения необходимо выполнить несколько операторов. На окончание условного оператора указывает **End If**.

Например:

```
IfA> 0 ThenC=A + B
MsgBox (C)
End If
```

Операторы **If** могут быть вложенными друг в друга, если необходимо проверить дополнительное условие в дополнении к исходному.

VB предоставляет сокращенную версию оператора **If .. Then .. Else**, являющуюся сжатым эквивалентом вложенных операторов –**If .. Then .. ElseIf**, с помощью которого можно проверить дополнительное условие.

Оператор перехода имеет вид **Go Тометка** и вызывает переход к выполнению оператора с указанной меткой или номером строки.

Например:

```
GoTo 123
. . .
123: Num = Num / 2
```

В алгоритмах обработки информации и реализующих их программах широко используются циклы – повторяющиеся одинаковые вычисления. Для реализации такого рода программ в *Visual Basic*используются операторы цикла.

Оператор цикла с параметром **For** используется в тех случаях, когда действия выполняются заданное число раз.

Синтаксис оператора следующий:

For параметр цикла = начальное значение To конечное значение [Step шаг]

операторы

[Exit For]

Next[счетчик цикла]

где For…Next – пара ключевых слов, определяющих начало и конец оператора цикла;

To – ключевое слово, разделяющее начальное значение и конечное значение параметра цикла;

Step – ключевое слово, используемое для задания шага цикла (необязательный);

ExitFor – ключевые слова, которые могут находиться в любом месте между For - Next и используются для прекращения выполнения цикла (необязательны);

параметр цикла – арифметическая переменная, не может быть элементом массива или элементом пользовательского типа данных.

Начальное значение, конечное значение и шаг определяют значения, которые принимает параметр цикла при работе программы. На первом шаге параметр цикла принимает начальное значение, после выполнения операторов, входящих в цикл, параметр цикла изменяется на величину шага, опять выполняются операторы, входящие в цикл, параметр цикла изменяется на величину шага и т. д., пока параметр цикла не примет последовательно все своизначения. После

того, как параметр цикла примет все свои значения и соответственное число раз выполнится блок операторов в цикле, будет выполняться следующий за Next оператор.

Операторы, входящие в цикла, будут выполняться, если:

- шаг цикла >= 0, то конечное значение >= начальное значение;
- шаг цикла < 0, то конечное значение <= начальное значение.

Например:

```
For  i = 1To  10
Sum = Sum + 2
Next i
```

Циклы с условием применяются тогда, когда некоторые действия в программе должны повторяться до тех пор, пока выполняется определенное условие, или до тех пор, пока не будет выполнено определенное условие. Т.е главной особенностью таких циклов является условие, которое может быть любым логическим выражением принимающим значение *true* или *false*. Циклы с условием получили название Do-Loop и делятся на циклы с предусловием и циклы с постусловием.

Синтаксис операторов цикла

Цикл DoWhile определяет выполнение операторов, входящих в цикл, до тех пор, пока логическое выражение принимает значение *true*.

Например, оператор цикла может быть использован для вычисления суммы чисел (Sum):

```
Dim Sum As Integer
Dim i As Integer
    Sum = 0 : i = 1
Do While Sum < 10
      Sum = Sum + 2 *
      i i = i + 1
Loop
Text1.Text = Sum
```

Цикл DoUntil определяет выполнение операторов, входящих в цикл, до тех пор, пока логическое выражение принимает значение *false*.

Например, для вычисления суммы чисел (Sum)можно использовать оператор цикла Do Until:

```
Dim Sum As Integer
Dim i As Integer
```

```
    Sum = 0 : i = 1
Do Until Sum > 9
      Sum = Sum + 2 * i
      i = i + 1
Loop
Text1.Text = Sum
```

Запись условия **While | Until логическое выражение** в начале или в конце цикла определяет, где это условие будет проверяться. Когда условие проверяется в начале цикла, цикл выполняется, если условие удовлетворено, т.е. значение логического выражения равно *true*. Такой вид цикла удобно применять в тех случаях, когда цикл не должен выполняться до тех пор, пока условие не будет выполнено.

Используя цикл с условием, можно организовать бесконечный цикл, когда логическое выражение либо никогда не выполняется, либо выполняется всегда. Выйти из бесконечного цикла и прервать работу программы можно, используя комбинацию клавиш *Ctrl + Break*.

При одновременном использовании в процедурах и функциях операторов цикла и условных операторов должно выполняться так называемое правило вложенности.

Если среди операторов, выполняющихся в цикле (циклы For-Next и Do-Loop), имеется условный оператор, то он должен целиком содержаться внутри цикла (между операторами For-Next или Do- Loop). Если в Then-блоке или Else-блоке условного оператора имеются выполняющиеся в цикле операторы, то эти циклы должны целиком содержаться в этих блоках.

В *Visual Basic* имеется возможность выделить неоднократно повторяющиеся фрагменты программы в определенным образом организованные функциональные блоки операторов, которые можно использовать без их повторного написания и отладки. При этом сложные программы получаются более компактными, а функциональные блоки при их правильной организации могут использоваться в других программных разработках. Организовать функциональные блоки в *Visual Basic* можно в виде процедур (Procedure) и функции (Function).

В программе, состоящей из нескольких процедур или функций, необходимо знать понятия локальных и глобальных данных.

Повторный вызов процедуры или функции внутри этой же процедуры или функции, обращенной к самой себе, называется **рекурсией**. *Visual Basic* допускает рекурсивное обращение.

Программный код процедур и функций, не являющихся процедурами об-

работки событий, помещается в раздел **general** списка **Object** формы. Кроме того, в раскрывающемся меню **Вставить** (Insert) главного меню имеется пункт Procedure, выбор которого раскрывает специальное окно для выбора процедуры или функции и задания имени. Для заданной процедуры или функции появляется шаблон в окне кода, первая и последняя строки программы, в который можно ввести текст программы.

Процедуры типа **Sub** вызываются по имени из процедуры обработки событий или из других процедур. Они также могут содержать аргументы, получать входные значения, выполнять определенные действия и возвращать значения.

Синтаксически процедура типа **Sub** определяется:

SubИмяПроцедуры (список аргументов)
операторы процедуры
[ExitSub]
End Sub

где Sub - EndSub – пара ключевыхслов, определяющих начало и конец процедуры;

имя процедуры – образуется в соответствии с общими правилами образования имен, но не может иметь описателей типа (имя процедуры не принимает значения);

ExitSub – ключевыеслова, которые используются для прекращения выполнения процедурыи выход из нее в вызывающую программу (необязательны).

При описании процедуры могут быть использованы следующие ключевые слова, которые записываются перед ключевым словом Sub и именем процедуры:

- Static определяет, что локальные данные (переменные, массивы), определенные внутри процедуры, сохраняются в промежутках между вызовами данной процедуры. Определение Static не распространяется на данные, которые объявлены вне данной процедуры, но используются в ней, т.е. эти данные будут изменяться независимо от определения Static. Нельзя использовать при рекурсивном вызове процедуры;
- Private определяет, что процедура может быть доступна только в том модуле, в котором она объявлена. Определение Private не имеет смысла, если процедура определена для какой-либо формы, так как в этом случае процедура недоступна из вне данной формы;
- Public определяет, что процедура может быть вызвана из любого модуля приложения.

Поскольку имя процедуры глобально определяется в рамках всех процедур всех модулей одной программы, имя процедуры не должно совпадать с другими глобально распознаваемыми именами программы.

Чтобы избежать конфликта в объявлении одних и тех же имен, можно ис-

пользовать определение Private, т.е. сделать процедуру доступной только в рамках модуля. В этом случае имя процедуры не может совпадать с такими именами, объявленными на уровне модуля, как имена переменных, массивов, констант и имена других процедур данного модуля.

Список аргументов имеет следующий синтаксис:
[Optional] [ByVal|ByRef] [ParamArray] ИмяАргумента1()AsТипАргумента1
[, ИмяАргумента2 () AsТипАргумента2]

Имя Аргумента – имя переменной, массива, элемента управления или формы. В случае массива используется имя **ИмяАргумента ( )** без указания границ значения индексов, что позволяет использовать одну процедуру для разного числа элементов массива в каждом конкретном случае.

Тип Аргумента может быть использован Byte, Boolean, Integer, Long, Currency, Single, Double, Date, String (только переменной длины). Variant, пользовательский тип или Control и Form. **As** необходимо применять для каждого аргумента.

Список аргументов определяет связь данных между вызывающей и вызываемой процедурами. Локальные данные, определенные внутри процедуры (кроме тех, что определены в списке general для форм), не могут использоваться в других процедурах. Процедуры, относящиеся к форме, но не являющиеся процедурами обработки событий, помещаются в раздел general формы.

Процедура не может быть определена внутри какой-либо процедуры или функции.

При описании аргументов процедуры могут быть использованы следующие ключевые слова, которые записываются перед именем аргумента:

- Optional определяет, что аргумент не является обязательным. При его использовании все последующие аргументы в списке аргументов также являются необязательными и для них также необходимо использовать данное ключевое слово. Аргументы, объявленные Optional, должны иметь тип Variant.
- ByVal определяет, что аргумент передается в процедуру по значению. В этом случае значение переменной не может быть модифицировано в ходе выполнения программы. Т.е. этим *Visual Basic* сообщается, чтобы он сохранил копию оригинала аргумента и вернул ее неизменной по окончании процедуры, даже если переменная была модифицирована в ходе выполнения процедуры.
- ByRef определяет, что аргумент перелается в процедуру по ссылке (используется по умолчанию). В этом случае значение переменной может быть модифицировано в ходе выполнения программы и уже измененным

возвращено в программу.

- ParamArray относится к последнему аргументу в списке аргументов и определяет, что конечный аргумент является необязательным массивом. Не может использоваться совместно с параметрами ByVal, ByRef, Optional.

Например:

```
PublicSubСравнение (E, X)
IfE = 0 ThenText4.Text "Ваш ответ не верный, возможно, не все данные ввели"
ElseIf  E = Val (Text3.Text) Then  Text4.Text = X
ElseText4.Text = «Ваш ответ неверный»
End If
End Sub
```

Чтобы использовать написанную подпрограмму, ее нужно вызвать.

Обычно подпрограмма вызывается из другой подпрограммы или функции с помощью специального оператора *VB*. Если она имеет аргументы, ей передается список фактических параметров.

Вызов процедуры Sub из другой процедуры можно произвести двумя способами:

1. **Имя Процедуры Список Фактических Параметров**
2. **Call Имя Процедуры (Список Фактических Параметров)**

Список Фактических Параметров – список аргументов, передаваемых процедуре. Он должен соответствовать списку, заданному в процедуре по количеству и типу. Если требуется использовать несколько процедур с одинаковыми названиями, при их вызове после имени процедуры через точку надо указывать имя модуля, на котором они расположены. А именно, Имя Модуля. Имя Процедуры Список Фактических Параметров.

В первом случае список фактических параметров задается без скобок, во втором – использование скобок обязательно. Но всегда список фактических параметров должен полностью соответствовать списку аргументов, заданному в объявлении подпрограммы. Все фактические параметры для обязательных аргументов должны быть перечислены в том порядке, в каком они присутствуют в описании подпрограммы, после чего могут идти параметры для необязательных аргументов.

Например: Call Сравнение (E, X).

VB позволяет вводить фактические параметры через имена аргументов в любом порядке и опускать необязательные (optional). При этом после имени аргумента ставятся двоеточие и знак равенства, после которого помещается зна-

чение аргумента (фактический параметр).

Функции (**Function**) в *Visual Basic* вызываются по имени из любой процедуры обработки события, связанного с объектом формы, или из других функций. Они не отличаются от встроенных функций, но разрабатываются пользователем. Функции могут содержать аргументы, через которые получают входные значения, а также возвращать значения в виде имени функции. Обычно они используются для вычислений.

Синтаксически функция определяется:

FunctionИмяФункции(списокаргументов)[AsТипФункции]
операторы функции
ИмяФункции = выражение
[Exit Function]
End Function

где Function - ExitFunction – пара ключевыхслов, определяющих начало и конец функции;

имя функции – образуется в соответствии с общими правилами образования имени может иметь описатель типа (**As Тип Функции** в конце первой строки относится к имени функции);

ExitFunction – ключевые слова, которые используются для прекращения выполнения функции и выход из нее в вызывающую программу (необязательны).

Имя функции принимает значение, и поэтому хотя бы один оператор **Имя Функции = выражение** должен присутствовать внутри функции и выполняться при выходе из нее. Если никакое значение не присвоено имени функции, то имя функции принимает значение по умолчанию: числовая функция принимает значение 0, функция, объявленная как String, принимает значение пустой строки нулевой длины, функция, объявленная как Variant принимает значение Empty.

Список аргументов имеет следующий синтаксис:

[Optional] [ByVal|ByRef][ParamArray]ИмяАргумента1 () AsТипАргумента1 [, ИмяАргумента2 () AsТипАргумента2]

Значения Имя Аргумента и Тип Аргумента определяются таким же образом, что и для процедур. Функция не может быть определена внутри какой-либо процедуры или функции. Изменяя значения аргументов, данную функцию можно использовать для работы с вектором произвольного размера. Используемые ключевые слова аналогичны рассмотренным для процедур.

Например:

```
Function СторонаТреугольника (Optional A, Optional B, Optional C)
  If  Not (IsMissing(A))  And  Not (IsMissing(B))
```

```
Then  СторонаТреугольника = Sqr(A ^ 2 + B ^ 2)
End If
  If  Not (IsMissing(A))  And  Not (IsMissing(C))
                          Then СторонаТреугольника = Sqr(C ^ 2 - A ^ 2)
End If
  If  Not (IsMissing(B))  And  Not (IsMissing(C))
                          ThenСторонаТреугольника = Sqr(C ^ 2 - B ^ 2)
EndIf
EndFunction
```

Вызов функции отличается от вызова процедуры. Обращение к функции является операндом выражения, т.е. в отличии от процедуры для ее вызова не используется специальный оператор, и имеет следующий синтаксис:

ИмяПеременной= ИмяФункции (список фактических параметров)

где список фактических параметров – это список значений аргументов, соответствующих списку аргументов в операторе Function по количеству, порядку следования и типу (если не используется параметр Optional).

Список фактических параметров при вызове функции должен обязательно заключаться в скобки.

Например:

```
Function Tangens(x)  As double
Tangens=sin(x)/cos(x)
endFunction
. . .
S = Tangens (1)
```

Список аргументов процедур и функций может включать имена форм и элементов управления (в качестве описателей типа для них применяются соответственно Form и Control). Это позволяет создавать универсальные алгоритмы для работы с формами и управляющими элементами.

Для контроля соответствия передаваемых при обращении типов форм и элементов управления в *Visual Basic* используется специальный оператор:

If TypeOf ИмяОбъекта Is ТипОбъекта Then . . .

где TypeOf и Is – ключевые слова, а последующий синтаксис и действие совпадают с условным оператором.

Для работы программ часто необходимо вводить исходные данные с клавиатуры, рабочего листа или текстовых файлов. Основные и промежуточные результаты работы программ выводятся на экран монитора, в ячейки рабочего листа таблицы *Excel*, окно отладки программы или в файлы на внеш-

них носителях.

Функция **MsgBox** выводит на экран диалоговое окно, содержащее сообщение, устанавливает режим ожидания нажатия кнопки пользователем, а затем возвращает значение типа Integer, указывающее, какая кнопка была нажата.

Синтаксис функции MsgBox:

MsgBox (prompt [, buttons] [, title] [, helpfile, context])

где prompt – строковое выражение, отображаемое как сообщение в диалоговом окне;

buttons – необязательный параметр целого типа, указывающий число и тип отображаемых кнопок, тип используемого значка, основную кнопку и модальность окна сообщения (по умолчанию равен 0);

title – необязательный параметр, являющийся заглавием диалогового окна, если этот аргумент опущен, в строку заголовка помещается имя приложения;

helpfile – необязательный параметр, являющийся строковым выражением, определяющим имя файла справки, содержащего справочные сведения о данном диалоговом окне, если этот параметр указан, необходимо указать также параметрcontext;

context – необязательный параметр, являющийся номером соответствующего раздела справочной системы, если этот параметруказан, необходимо указать также параметр helpfile.

Функция MsgBox содержит один обязательный параметр (prompt) и четыре необязательных.

Максимальная длина строки prompt составляет приблизительно 1024 символов и зависит от ширины используемых символов. Строковое значение prompt может содержать нескольких физических строк.

Значения переменной buttons можно задавать в виде целого числа, именованной константы или арифметического выражения. В табл. 14 приведены значения параметра buttons, имена всех поименованных констант и в скобках – их значение.

Если указаны оба параметра helpfile и context, то пользователь имеет возможность нажатием клавиши *F*1 вызвать контекстную справку.

Функцию MsgBox с двумя или большим числом аргументов можно использовать только в выражении. Наличие запятых, соответствующих отсутствующим параметрам, является обязательным.

Например:

```
PrivateSubForm_Load()
Dim rv As Long
rv = MsgBox("5 + 5 = 10", vbYesNo Or vbQuestion)
Ifrv = vbYesThenMsgBox "Правильно" ElseMsgBox "Неправильно", vbExclamation
```

```
End If
EndSub
```

При обращении к функции MsgBox с одним параметром допускается упрощенный вызов функции без использования возвращаемого значения.

Функция InputBox выводит на экран диалоговое окно, содержащее сообщение и поле ввода, устанавливает режим ожидания ввода текста пользователем или нажатия кнопки, а затем возвращает значение типа String, содержащее текст, введенный в поле.

Синтаксис функции InputBox:

InputBox(Prompt[, Title] [, Default] [, Xpos] [, Ypos] [, Helpfile, Context])

где prompt –строковое выражение, отображаемое как сообщение в диалоговом окне;

title – необязательный параметр, являющийся строковым выражением, отображаемым в строке заголовка диалогового окна, если этот параметр опущен, в строку заголовка помещается имя приложения;

default – строковое выражение, отображаемое в поле ввода как используемое по умолчанию, если пользователь не введет другую строку, если этот параметр опущен, поле ввода изображается пустым;

xpos – числовое выражение, задающее расстояние по горизонтали между левой границей диалогового окна и левым краем экрана (в твипах), если этот параметр опущен, диалоговое окно выравнивается по центру экрана по горизонтали;

ypos – числовое выражение, задающее расстояние по вертикали междуверхней границей диалогового окна и верхним краем экрана, если этот аргумент опущен, диалоговое окно помещается по вертикали примерно на одну треть высоты экрана;

helpfile – необязательный параметр, являющийся строковым выражением, определяющим имя файла справки, содержащего справочные сведенияо данном диалоговом окне, если этот параметр указан, необходимо указать также параметрcontext;

context – необязательный параметр, являющийся номером соответствующего раздела справочной системы, если этот параметруказан, необходимо указать также параметр helpfile.

Функция InputBox содержит один обязательный параметр (prompt) ишесть необязательных.

Таблица 14

Значения параметра buttons

Значения	Описание
Допустимые значения	
vbOKOnly (0)	только кнопка OK
vbOKCancel (1)	кнопки OK и Отмена (Cancel)
vbAbortRetryIgnore(2)	кнопки Прервать (Abort), Повторить (Retry) и Пропустить (Ignore)
vbYesNoCancel (3)	кнопки Да (Yes), Нет (No) и Отмена (Cancel)
vbYesNo (4)	кнопки Да (Yes) и Нет (No)
vbRetryCancel (5)	кнопки Повторить (Retry) и Отмена (Cancel)
vbCritical (16)	используется значок «Критическое сообщение»
vbQuestion (32)	используется значок «Предупреждающий запрос»
vbExclamation (48)	используется значок «Предупреждение»
vbInformation (64)	используется значок «Информационное сообщение»
vbDefaultButton1 (0)	основной является первая кнопка
Возвращаемые значения	
vbOK (1)	нажата кнопка *OK*
vbCancel (2)	нажата кнопка Отмена (Cancel)
vbAbort (3)	нажата кнопка Прервать (Abort)
vbRetry (4)	нажата кнопка Повторить (Retry)
vbIgnore (5)	нажата кнопка Пропустить (Ignore)
vbYes (6)	нажата кнопка Да (Yes)
vbNo (7)	нажата кнопка Нет (No)

Функцию InputBox с двумя или большим числом параметров можно использовать только в выражении. Наличие запятых, соответствующих отсутствующим аргументам, является обязательным.

Например:

```
SubТестНаФункциюInputBox()
Dim Message As String, Title As String, Default As String, MyValue As String
Message = "Введите число от 4 до 7"
Title = "Пример использования функции InputBox"
Default = "5"
MyValue = InputBox(Message, Title, Default)
MsgBox( MyValue )
MyValue = InputBox(Message, Title, Default, 100, 100)
MsgBox( MyValue )
End Sub
```

Функция InputBox вводит текстовую информацию. В случае если необходимо ввести числовую информацию, применяется функция Val, которая преобразует переменные типа String в Double.

Вопрос 5. Технология офисного программирования в среде VBA

VBA - немного упрощённая реализация языка программирования Visual Basic, специально разработанная и встроенная в линейку продуктов Microsoft Office и другие программные пакеты, такие как: AutoCAD, SolidWorks, CorelDRAW, WordPerfect).

Visual Basic for Application (VBA) является современным языком визуального и объектно-ориентированного программирования и предназначен для расширения функциональных возможностей документа.

Отличительной особенностью VBA является использование обычных переменных и констант с имеющимися объектами приложений MS Office.

Например, в MS Office Excel 2010 это могут быть рабочие книги, рабочие листы, диапазоны ячеек, диаграммы и т. д.

Также можно разрабатывать приложения, которые включают различные компоненты нескольких приложений Office и способствуют тем самым интеграции и совместному использованию данных.

Интегрированная среда разработки (редактор VBA) представляет собой отдельное приложение, запускающееся только в программах MS Office (Рис.47).

Модули VBA, т.е. место, где хранится код макроса на языке VBA, сохраняются в файлах MS Office.

Запуск редактора VBA:

- Пуск – Все Программы – MS Office – MS Excel
- на вкладке Разработчик в группе Код кнопка Visual Basic
- или комбинация клавиш ALT + F11

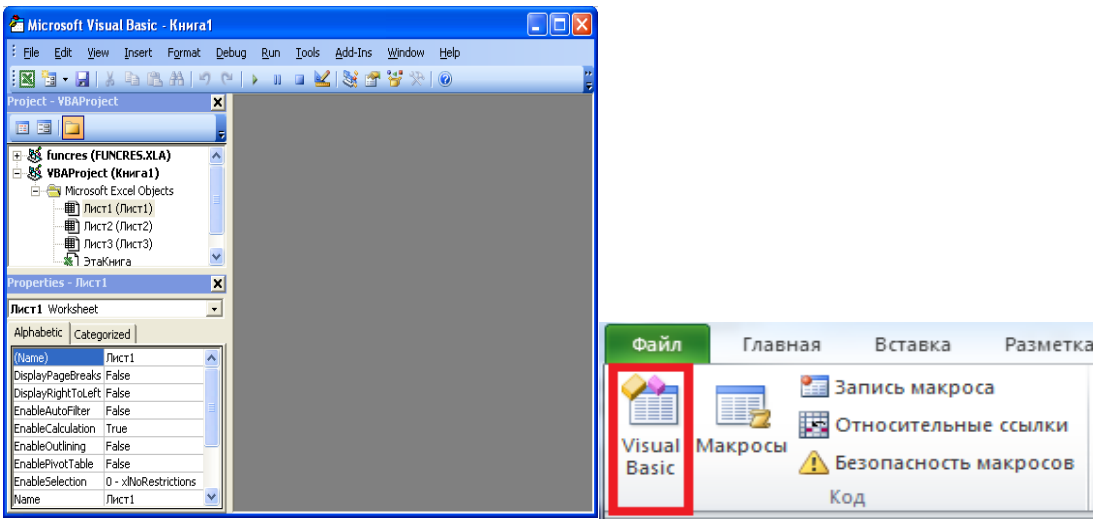


Рис. 47. Интегрированная среда разработки VBA

Интерфейс редактора VBA (Рис.48) состоит из компонентов:

- строка заголовка
- строка меню
- строка инструментов
- окно проекта
- окно свойств
- окно редактирования форм
- плавающая панель инструментов
- окно редактирования кода

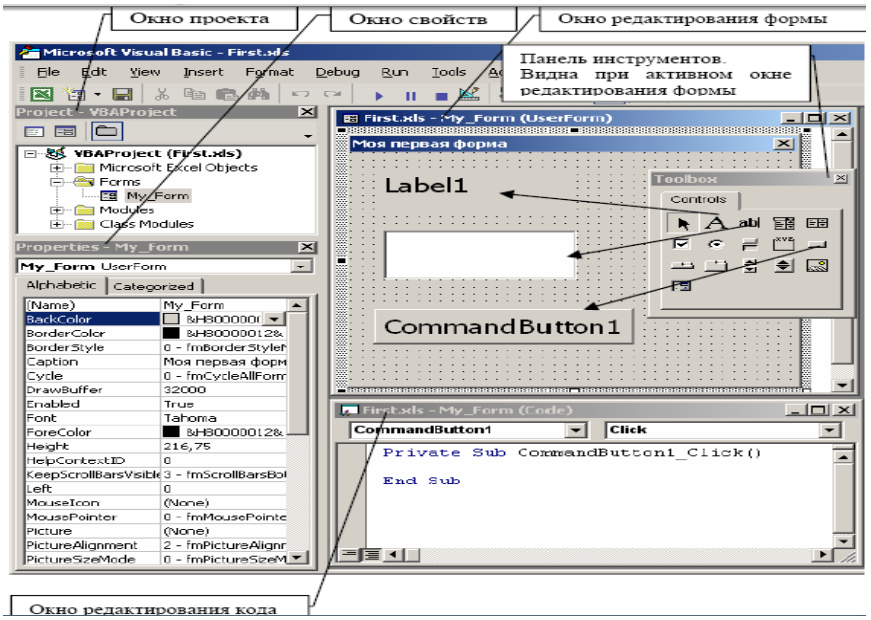


Рис. 48. Интерфейс редактора VBA

Основные компоненты редактора VBA

- Окно проводника проекта (Project)
- Окно свойств (Properties)
- Окно редактирования форм (UserForm)
- Панель элементов управления (ToolBox)
- Окно редактирования кода (Code)

Окно проводника проекта (Project Explorer)

В нем отображается иерархическая структура всех открытых проектов и их составляющих: файлов форм и модулей (Рис.49).

Окно проводника проекта можно вызвать:

- Меню – View – Project Explorer
- Панель инструментов – кнопка
- Комбинацией клавиш Ctrl+R

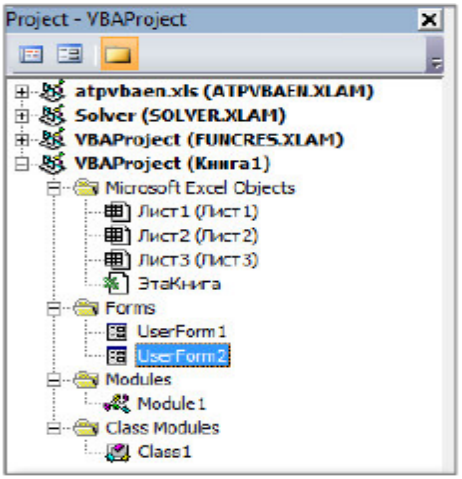


Рис. 49. Окно проводника проекта

Панель инструментов окна проводника содержит 3 кнопки (Рис.50):

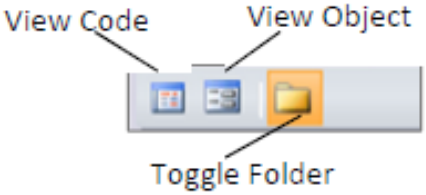


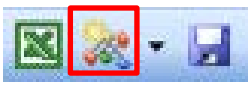
Рис. 50. Панель инструментов окна

- кнопка View Code (показать программный код), выводит на экран окно программного кода для выделенного объекта;
- кнопка View Object (показать объект), выводит на экран поверх всех окон сам выделенный объект;
- кнопка Toggle Folder (переключение отображения папок), управляет режимом отображения в самом окне проектов папок среднего уровня иерархии структуры файлов и модулей проектов.

Добавление Формы в проект:

- команда Insert-UserForm;
- панель инструментов – кнопка 

Добавление Модуля в проект:

- команда Insert-Module;
- панель инструментов – кнопка 

Удаление формы или модуля из проекта производится выбором соответствующего значка с последующим выбором команды **File-Remove**.

Окно свойств объекта (Properties)

Это окно определяют внешний вид объекта, его поведение и изменяет

характеристики, т.е. просматривает и изменяет свойства выделенного объекта (проекта, модуля, формы, элемента управления) (Рис.51).

Свойства можно отображать как в алфавитном порядке, так и по категориям, посредством выбора соответствующих вкладок Alphabetic или Categorized.

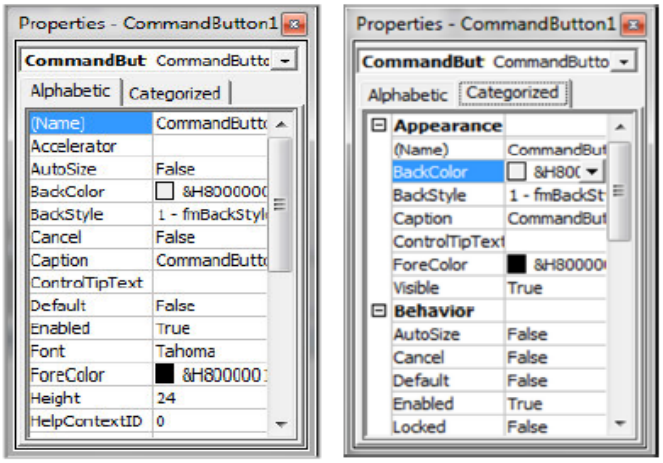


Рис. 51. Окно свойств объекта

Окно свойств можно вызвать:

- Меню – *View – Properties Window* ;
- Панель инструментов – кнопка 
- Клавиша F4

Окно редактирования форм (UserForm)

Графический интерфейс необходим для реализации интерактивного диалога пользователя с приложением. Основой для создания графического интерфейса разрабатываемого приложения является **форма**, в которой выполняется проектирование приложения (Рис.52).

Форма – это объект, представляющий собой пустое диалоговое окно на экране, в котором размещаются управляющие элементы.

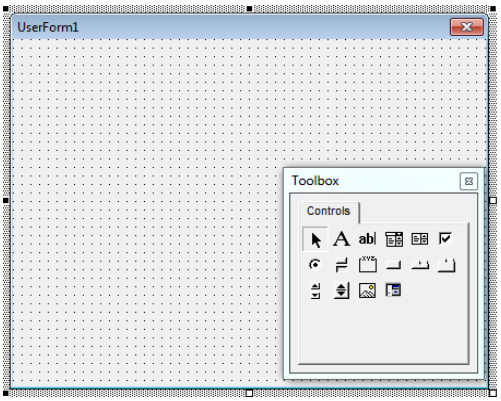
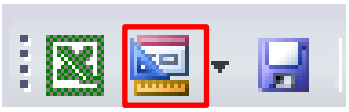


Рис. 52. Окно редактирования форм

Вставка новой формы UserForm:

- Insert (Вставка) → UserForm
- Панель инструментов – кнопка 

Существует другие способы открыть окно требуемой формы:

- окно проектов – 1щ ПКМ на форме - КМ - View Object;
- в окне проектов выделите имя формы, окно которой нужно открыть – Меню - View Object или Shift + F7;
- в окне проектов 2щ ЛКМ на форме.

В рабочей книге может быть произвольное количество форм UserForm, а каждая форма включает единственное пользовательское диалоговое окно.

Размер формы можно изменять, используя маркеры изменения размера.

Панель элементов управления (ToolBox)

Панель ЭУ – это основной рабочий инструмент разработки форм приложения (Рис.53).

Элемент управления – это специализированный объект, который можно размещать на формах VBA (или непосредственно в рабочей книге) и который используется для организации взаимодействия с пользователем.

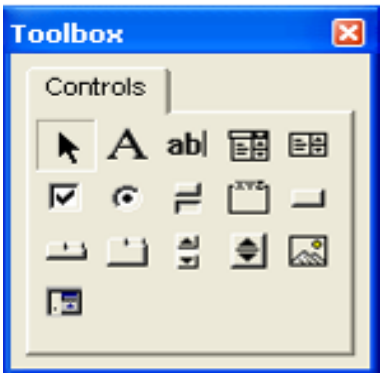


Рис. 53. Панель элементов управления

Панель элементов управления вызывается:

- Меню – View – Toolbox
- Панель инструментов – кнопка 

В составе Панели элементов управления содержатся основные элементы управления форм – метки, текстовые поля, кнопки и другие элементы для быстрого визуального проектирования макета формы.

Добавить элементы управления на форму можно:

- перетащить кнопку из панели Toolbox на форму;
- активизировать кнопку на панели Toolbox и построить ее на форме.

Окно редактирования кода (Code)

Окно выполняет функции текстового редактора для ввода и изменения процедур и функций проекта. Каждому объекту в проекте соответствует свое окно кода (Рис.54).

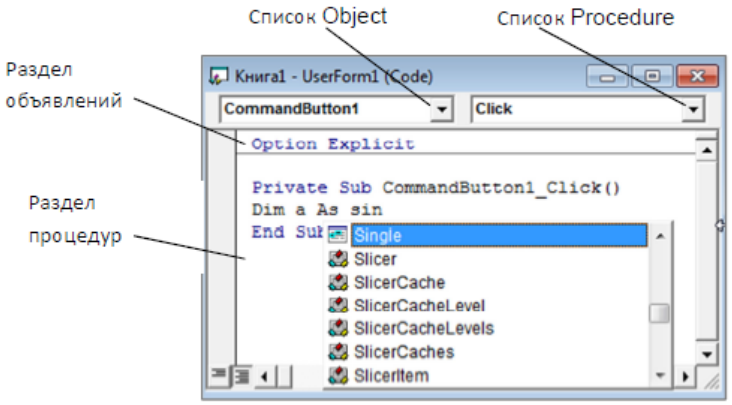


Рис. 54. Окно редактирования кода

Окно редактирования кода можно открыть:

- Меню – View – Code
- 2щ ЛКМ на выбранном элементе управления на форме
- Окно проводника проекта – пиктограмма – View Code
- Нажать клавишу F7
- Использовать Контекстное меню.

Редактор кода позволяет автоматизировать написание программных операторов, свойств и параметров (Рис.55).

При вводе предлагается список компонентов, логически завершающих вводимую конструкцию.

Например, при наборе кода после слова на экране отобразится список компонентов, которые логически могут завершить эту конструкцию.

Редактор кода автоматически отображает на экране сведения о процедурах, функциях, свойствах и методах после набора их имени.

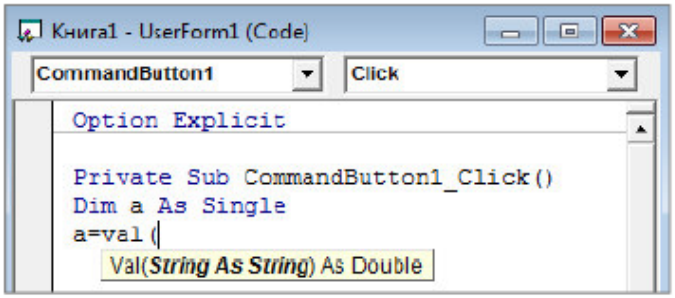


Рис. 55. Редактор кода

ТЕМА 8. КОМПЬЮТЕРНЫЕ СЕТИ

План:

1. Основы построения компьютерных сетей.
2. Интернет-технология.
3. Гипертекстовая и мультимедиа технологии.

Вопрос 1. Основы построения компьютерных сетей

В 1960-е годы появились первые вычислительные сети с ЭВМ. С этого времени собственно и появляются сетевые информационные технологии, позволившие объединить технологии сбора, хранения, передачи и обработки информации на ЭВМ с техникой связи

В настоящее время становиться все труднее представить себе компьютер, не имеющий доступа к сети. Даже единственному домашнему компьютеру на практике все чаще доступно сетевое подключение, по крайней мере, через модем. Возникла целая категория пользователей, для которых компьютер в первую очередь служит средством доступа к интернету.

Одной из главных объективных причин развития компьютерных сетей является усложняющийся характер повседневных производственных задач, которые приходится решать людям. Как следствие, они решаются коллективно, а также с применением большого количества информации. Постоянная необходимость оперативного информационного обмена в процессе работы над общим проектом или использование всевозможных распределенных ресурсов служат лучшим катализатором для развития и совершенствования сетевых технологий.

Компьютерная сеть – это совокупность компьютеров, соединенных с помощью каналов связи и средств коммутации в единую систему для обмена сообщениями и доступа пользователей к различным ресурсам.

Сложные сети подразумевают большое количество пользователей, разветвленную структуру, узлы коммутации и коммуникации, соединяющие всех в единую структуру.

Основу сетевых технологий составляют вычислительные сети – средства связи (телекоммуникации), с помощью которых распределенные в пространстве компьютеры объединяются в систему.

Вычислительную сеть называют также **сетью ЭВМ** или **компьютерной сетью** (*Computer network*). Она представляет вычислительный комплекс, включающий территориально распределенную систему компьютеров и их терминалов, объединенных в единую систему. Почти сразу же с появлением вычислительных сетей они стали использоваться для обмена различного рода данными

(сети передачи данных) и информацией. Развитие компьютерных сетей и сетевых технологий показало возможность с их помощью организовать широкомасштабное информационное обеспечение людей. Это привело к тому, что вычислительные сети, обеспечивающие обмен информационными ресурсами, стали называть «информационными сетями», представляя разновидность коммуникационных сетей.

Коммуникационная сеть предназначена для передачи данных, также она выполняет задачи, связанные с преобразованием данных. Коммуникационные сети различаются по типу используемых физических средств соединения.

Информационная сеть предназначена для хранения информации и состоит из информационных систем. На базе коммуникационной сети может быть построена группа информационных сетей. Под информационной системой следует понимать систему, которая является поставщиком или потребителем информации.

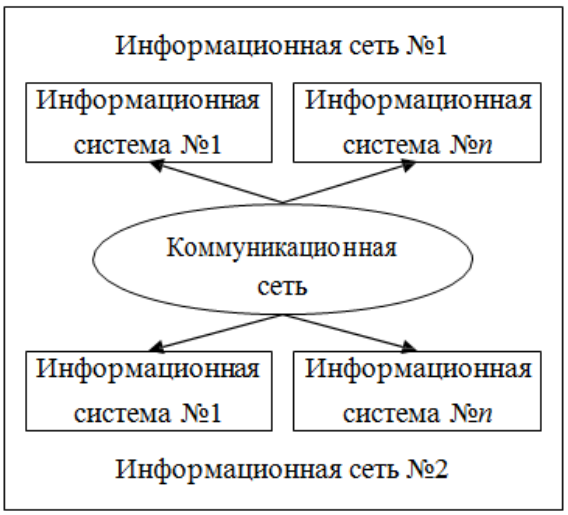


Рис. 56. Информационные и коммуникационные сети

Компьютерная сеть состоит из информационных систем и каналов связи. Под **информационной системой** следует понимать объект, способный осуществлять хранение, обработку или передачу информации. В состав информационной системы входят: компьютеры, программы, пользователи и другие составляющие, предназначенные для процесса обработки и передачи данных. **Коммуникационная сеть** – это система объектов, осуществляющих функции создания (генерации), преобразования, хранения и потребления продукта и линий передачи, по которым осуществляется передача этого продукта внутри сети.

Сети обычно находятся в частном ведении пользователя и занимают некоторую территорию. По области использования (распространения) выделяют локальные, региональные (территориальные) и глобальные сети.

Локальные вычислительные сети (ЛВС) или *Local Area Network* (*LAN*), расположенные в одном или нескольких близко расположенных зданиях. ЛВС

обычно размещаются в рамках какой-либо организации (корпорации, учрежде ния), поэтому их называют корпоративными. Они имеют по сравнению с *WAN* менее развитую архитектуру и используют более простые методы управления взаимодействием узлов сети. Небольшие расстояния между узлами сети и простота управления системой связи позволяют обеспечивать высокую скорость передачи данных.

В ЛВС расстояние между компьютерами обычно ограничено до 1–2,5 км, скорость передачи информации составляет более одного Мбит/с. Такая сеть состоит из трех основных компонент: одной или нескольких центральных (главных) машин (серверов), рабочих станций и коммуникаций.

Территориальные сети – (*Wide Area Network, WAN*) сети, расположенные в разных зданиях, городах и странах и охватывающие различные географические пространства, делят на региональные и глобальные сети.

Региональные сети обычно охватывают административную территорию города, области и т.п., а также производственные и иные объединения, расположенные в нескольких районах города, нескольких городах и т.п. К региональным относятся корпоративные сети (сети масштаба предприятия), связывающие между собой локальные вычислительные сети, охватывающие территорию, как правило, представляющую одно или несколько близко расположенных зданий, входящих в состав этой корпорации (предприятия).

Глобальная вычислительная сеть (*Wide Area Network, WAN*) – это множество географически удаленных друг от друга компьютеров (*host*-узлов), взаимодействующих между собой с помощью коммуникационных каналов передачи данных и специального программного обеспечения – сетевых операционных систем. Хост-компьютеры – мощные многопользовательские вычислительные системы (сервера), а также специализированные компьютеры, выполняющие функции коммуникационных узлов. Пользователи персональных компьютеров становятся абонентами такой сети после подключения своих компьютеров к ее основным узлам.

В качестве примеров распределенных сетей очень большого масштаба можно назвать: *Internet, EUNET, Relcom, FIDO*.

Интернет – глобальная информационная сеть, состоящая из большого количества сетей различного назначения, выполняющих разные задачи. Интернет образует интегрированную информационную сеть (интерсеть) – совокупность расположенных в различных странах взаимосвязанных информационных сетей, называемых подсетями.

Современные сетевые технологии представляют возможность работать в отложенном (оффлайн) и интерактивном (онлайн) режиме, обеспечивают связь с любыми доступными информационными источниками, позволяют осуществ-

лять профессионально-ориентированное консультирование и обучение и др.

Онлайновые технологии (*On line*) – это средства коммуникации сообщений в сетевом информационном пространстве, обеспечивающие синхронный обмен информацией в реальном времени. Онлайновые технологии включают интерактивные виды услуг в интернете: *ISQ*, интернет-телефонию и др.

Оффлайновые технологии (*Off line*) – это средства электронной коммуникации сообщений в сетевом информационном пространстве, допускающие существенную асинхронность в обмене данными и сообщениями. Оффлайновые технологии включают: списки рассылки, группы новостей, веб-форумы, электронную почту и т.д.

Преимущества, предоставляемые компьютерными сетями по сравнению с отдельно работающими компьютерами:

1. Использование аппаратных ресурсов более мощного или лучше оснащенного, чем у вас, компьютера. Например, с помощью обращения по сети можно выполнить сложную вычислительную задачу на машине с более производительным процессором или отпечатать документ на цветном лазерном принтере, установленном в соседней комнате.

2. Использование программы, работающей на доступном по сети компьютере. На этом принципе функционируют многие крупные компьютеры-серверы, которые, получая запросы по сети, мгновенно проводят сложную обработку и возвращают в качестве ответа результаты. Иногда подобное разделение программного обеспечения организовано в виде совместного использования общей копии программы, установленной на одной из машин.

3. Использование данных, хранящихся на удаленных компьютерах. Ярким примером такого рода является доступ к информации через интернет; другим примером может служить работа с распределенными базами данных.

4. Обмен информацией между пользователями компьютеров, совместная робота над общими проектами. Наиболее наглядным (но далеко не единственным) примером такого рода является электронная почта.

Недостатки компьютерных сетей:

1. Подключение компьютера к сети делает потенциально возможным несанкционированный доступ извне к находящейся на нем информации.

2. Активное использование сетей для распространения компьютерных вирусов.

3. Чрезмерное увлечение виртуальным общением несет определенную опасность для отдельных людей и общества в целом.

В информационных сетях управляющие системы называются серверами. Под термином **сервер** (англ. *server* – обслуживающий процессор, узел обслуживания) понимают подключенную к сети достаточно мощную вычислитель-

ную машину, обладающую определенными ресурсами общего пользования, а также, как правило, возможностью объединять некоторое количество компьютеров как в локальной, так и в глобальной информационных сетях. Сетевые узлы с серверами, называют хостами (англ. *host* – хозяин). Обычно они становятся провайдерами интернета.

Серверы обычно выполняют функции административного управления в сети и при этом называются администраторами системы. В их задачи входит проверка работоспособности системы (каналов, компьютеров, программ и т.п.); выявление сбоев, несанкционированного доступа и других нарушений в сети; восстановление работоспособности сети; учет работы сети, подготовка отчетов о ее работе и предоставление пользователям информации о ресурсах сети.

По назначению серверы делятся на: файловый, коммуникационный, приложений, почтовый и др. Кроме того, в сетях используют: сервер баз данных (*Data Base Server*), принт-сервер, факс-сервер и др.

Подключенные в сети к серверам компьютеры, предназначенные для решения задач пользователя, называют рабочими станциями или клиентами (*client*). Разница заключается в применяемом программном обеспечении, позволяющем использовать компьютеры в сети только как сервер или как *PC*. Возможен вариант, когда любой компьютер в сети может быть в одних условиях сервером, а в других – клиентом. **Клиентом** обычно считается менее мощный компьютер, ресурсы которого не предоставляются в совместное использование в сети.

Состав основных элементов в сети зависит от ее архитектуры.

Архитектура сети – это концепция, определяющая взаимосвязь, структуру и функции взаимодействия рабочих станций в сети. Она предусматривает логическую, функциональную и физическую организацию технических и программных средств сети. Архитектура определяет принципы построения и функционирования аппаратного и программного обеспечения элементов сети.

В основном выделяют три вида архитектур:

- архитектура терминал – главный компьютер;
- одноранговая архитектура;
- архитектура клиент-сервер.

Архитектура терминал – главный компьютер (*terminal – host computer architecture*) – это концепция информационной сети, в которой вся обработка данных осуществляется одним или группой главных компьютеров.

Рассматриваемая архитектура предполагает два типа оборудования. Главный компьютер, где осуществляется управление сетью, хранение и обработка данных. Терминалы, предназначенные для передачи главному компьютеру команд на организацию сеансов и выполнения заданий, ввода данных для выполне-

ния заданий и получения результатов. Главный компьютер через мультиплексоры передачи данных (МПД) взаимодействуют с терминалами, как представлено на рис. 57. Классический пример архитектуры сети с главными компьютерами – системная сетевая архитектура (*System Network Architecture – SNA*).

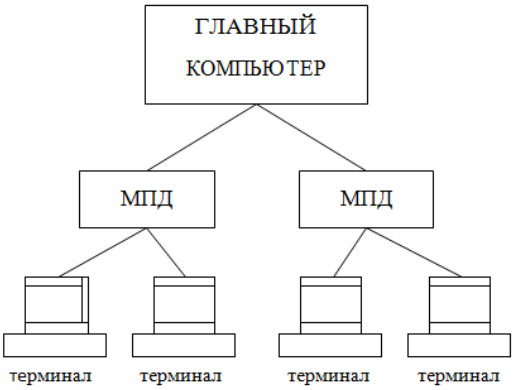


Рис. 57. Архитектура терминал – главный компьютер

Одноранговая архитектура (*peer-to-peer architecture*) – это концепция информационной сети, в которой ее ресурсы рассредоточены по всем системам. Данная архитектура характеризуется тем, что в ней все системы равноправны. К одноранговым сетям относятся малые сети, где любая рабочая станция может выполнять одновременно функции файлового сервера и рабочей станции. В одноранговых локальных вычислительных сетях дисковое пространство и файлы на любом компьютере могут быть общими. Чтобы ресурс стал общим, его необходимо отдать в общее пользование, используя службы удаленного доступа сетевых одноранговых операционных систем. В зависимости от того, как будет установлена защита данных, другие пользователи смогут пользоваться файлами сразу же после их создания. Одноранговые локальные вычислительные сети достаточно хороши только для небольших рабочих групп.

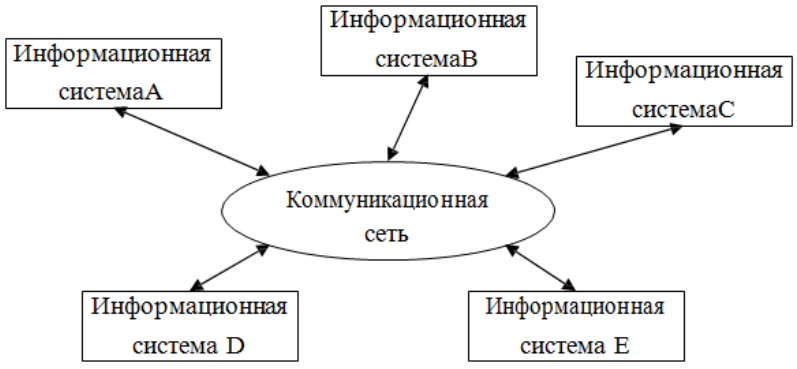


Рис. 58. Одноранговая архитектура

Одноранговые сети имеют следующие преимущества:

- они легки в установке и настройке;
- отдельные ПК не зависят от выделенного сервера;

- пользователи в состоянии контролировать свои ресурсы;
- малая стоимость и легкая эксплуатация;
- минимум оборудования и программного обеспечения;
- нет необходимости в администраторе;
- подходят для сетей с количеством пользователей, не превышающим десяти.

Проблемой одноранговой архитектуры является ситуация, когда компьютеры отключаются от сети. В этих случаях из сети исчезают виды сервиса, которые они предоставляли. Сетевую безопасность одновременно можно применить только к одному ресурсу, и пользователь должен помнить столько паролей, сколько сетевых ресурсов. При получении доступа к разделяемому ресурсу ощущается падение производительности компьютера. Существенным недостатком одноранговых сетей является отсутствие централизованного администрирования.

Использование одноранговой архитектуры не исключает применения в той же сети других архитектур.

Архитектура клиент-сервер (*client-server architecture*) – это концепция информационной сети, в которой основная часть ее ресурсов сосредоточена в серверах, обслуживающих своих клиентов (рис. 59). Рассматриваемая архитектура определяет два типа компонентов: серверы и клиенты.

Сервер работает по заданиям клиентов и управляет выполнением их заданий. После выполнения каждого задания сервер посылает полученные результаты клиенту, пославшему это задание.

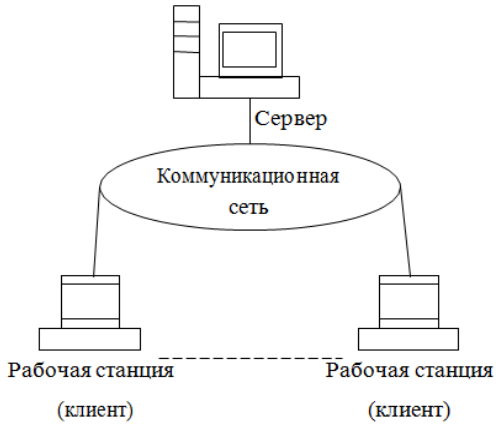


Рис. 59. Архитектура клиент-сервер

Сервисная функция в архитектуре клиент-сервер описывается комплексом прикладных программ, в соответствии с которым выполняются разнообразные прикладные процессы. Процесс, который вызывает сервисную функцию с помощью определенных операций, называется клиентом. Им может быть программа или пользователь.

Клиенты – это рабочие станции, которые используют ресурсы сервера и

предоставляют удобные интерфейсы пользователя. Интерфейсы пользователя – это процедуры взаимодействия пользователя с системой или сетью. Клиент является инициатором и использует электронную почту или другие сервисы сервера. В этом процессе клиент запрашивает вид обслуживания, устанавливает сеанс, получает нужные ему результаты и сообщает об окончании работы.

Сети на базе серверов имеют лучшие характеристики и повышенную надежность.

Помимо сетевой операционной системы необходимы сетевые прикладные программы, реализующие преимущества, предоставляемые сетью.

Наиболее распространенные сетевые операционная системы – *NetWare* фирмы *Novel*; *WindowsNT* фирмы *Microsoft*; *UNIX* фирмы *AT&T*; *Linux*.

В современной клиент-серверной архитектуре выделяется четыре группы объектов: клиенты, серверы, данные и сетевые службы. Клиенты располагаются в системах на рабочих местах пользователей. Данные в основном хранятся в серверах. Сетевые службы являются совместно используемыми серверами и данными. Кроме того службы управляют процедурами обработки данных.

Сети клиент-серверной архитектуры имеют следующие преимущества:

- позволяют организовывать сети с большим количеством рабочих станций;
- обеспечивают централизованное управление учетными записями пользователей, безопасностью и доступом, что упрощает сетевое администрирование;
- эффективный доступ к сетевым ресурсам;
- пользователю нужен один пароль для входа в сеть и для получения доступа ко всем ресурсам, на которые распространяются права пользователя.

Наряду с преимуществами сети клиент – серверной архитектуры имеют и ряд недостатков:

- неисправность сервера может сделать сеть неработоспособной, как минимум потерю сетевых ресурсов;
- требуют квалифицированного персонала для администрирования;
- имеют более высокую стоимость сетей и сетевого оборудования.

Для подсоединения компьютера к сети необходимо наличие специальных аппаратных и программах средств. К аппаратным средствам компьютерных сетей относится несколько видов устройств:

- сетевые компьютеры (оснащенные сетевым адаптером);
- каналы связи (кабельные, спутниковые, телефонные, цифровые, волоконно-оптические, радиоканалы и др.);
- различного рода преобразователи сигналов;
- сетевое оборудование.

Для организации передачи и приема данных каждый из компьютеров должен иметь специальное сетевое устройство, которое может быть двух типов:

сетевой адаптер и модем. Разница между ними заключается в том, что **сетевой адаптер** организует работу по специализированным компьютерным линиям связи, а **модем** – по обычной телефонной сети или иным уже имеющимся линиям, например, сети кабельного телевидения. Карты сетевых адаптеров устанавливаются на каждой рабочей станции и на файловом сервере. Рабочая станция отправляет запрос через сетевой адаптер к файловому серверу и получает ответ через сетевой адаптер, когда файловый сервер готов. Сетевые адаптеры вместе с сетевым программным обеспечением способны распознавать и обрабатывать ошибки, которые могут возникнуть из-за электрических помех, коллизий или плохой работы оборудования. Характерной особенностью применением модемов является временный (для осуществления передачи данных по аналоговой линии) переход от цифровой формы представления к аналоговой.

Для доставки данных от одного компьютера к другому используются каналы связи. Под **каналом связи** (*data link*) следует понимать путь или средство, по которому передаются сигналы. Средство передачи сигналов называют абонентским, или физическим, каналом.

Каналы связи создаются по линиям связи при помощи сетевого оборудования и физических средств связи. Физические средства связи построены на основе витых пар, коаксиальных кабелей, оптических каналов или эфира. Между взаимодействующими информационными системами через физические каналы коммуникационной сети и узлы коммутации устанавливаются логические каналы.

Логический канал – это путь для передачи данных от одной системы к другой. Логический канал прокладывается по маршруту в одном или нескольких физических каналах. Логический канал можно охарактеризовать как маршрут, проложенный через физические каналы и узлы коммутации.

Для организации связи в сетях используются следующие понятия:

- кабели связи;
- линии связи;
- каналы связи.

Кабель связи – это длинномерное изделие электротехнической промышленности. Из кабелей связи и других элементов (монтаж, крепеж, кожухи и т.д.) строят линии связи. Прокладка линии внутри здания задача достаточно серьезная. Длина линий связи колеблется от десятков метров до десятков тысяч километров. В любую более-менее серьезную линию связи кроме кабелей входят: траншеи, колодцы, муфты, переходы через реки, море и океаны, а также грозозащита (равно как и другие виды защиты) линий. Очень сложны охрана, эксплуатация, ремонт линий связи; содержание кабелей связи под избыточным давлением, профилактика (в снег, дождь, на ветру, в траншее и в колодце, в ре-

ке и на дне моря). Большую сложность представляют собой юридические вопросы, включающие согласование прокладки линий связи, особенно в городе.

По уже построенным линиям организуют каналы связи. Причем если линию, как правило, строят и сдают сразу всю, то каналы связи вводят постепенно. Уже по линии можно дать связь, но такое использование крайне дорогостоящих сооружений очень неэффективно. Поэтому применяют аппаратуру каналообразования (или, как раньше говорили, уплотнение линии). По каждой электрической цепи, состоящей из двух проводов, обеспечивают связь не одной паре абонентов (или компьютеров), а сотням или тысячам: по одной коаксиальной паре в междугородном кабеле может быть образовано до 10800 каналов тональной частоты (0,3 – 3,4 КГц) или почти столько же цифровых, с пропускной способностью 64 Кбит/с.

Классическим вариантом канала является использование качественных электрических кабелей. В качестве среды передачи данных используются различные виды кабелей: коаксиальный кабель (центральная токопроводящая жила и вокруг нее экранирующая металлическая оплетка), кабель на основе экранированной и неэкранированной витой пары (свитые вместе провода) и оптоволоконный кабель (световоды, информацию по которым переносит световой луч).

Типичные характеристики для трех основных типов кабеля могут быть сведены в следующую таблицу.

Таблица 15

Характеристики для трех основных типов кабеля

Показатели	Среда передачи данных		
	Коаксиал	Витая пара	Оптоволокно
Цена	Невысокая	Невысокая	Высокая
Наращивание	Простое	Очень простое	Проблематичное
Ограничение длинны	1,5 км	300 – 500 м	50 км
Скорость обмена	До 300 Мбит/с	10 – 100 Мбит/с	Гбиты/с
Защита от прослушивания	Хорошая, но легко отвлекается	Незначительная	Высокая
Восприимчивость к помехам	Существует	Существует	Отсутствует
Проблемы с заземлением	Возможны	Нет	Нет

Наиболее популярным видом среды передачи данных на небольшие расстояния становится неэкранированная витая пара. Оптоволоконный кабель широко применяется как для построения локальных связей, так и для образования магистралей глобальных сетей. Оптоволоконный кабель может обеспечить очень высокую пропускную способность канала и передачу на значительные расстояния.

Кабель типа «витая пара» (*twisted pair*) - изолированная пара проводников

скручена с небольшим числом витков на единицу длины. Скручивание проводов уменьшает электрические помехи извне при распространении сигналов по кабелю, а экранированные витые пары еще более увеличивают степень помехозащищенности сигналов. Кабель типа «витая пара» используется во многих сетевых технологиях, включая *Ethernet, ARCNet* и *IBMToken Ring*.

Коаксиальные кабели используются в радио и телевизионной аппаратуре. Они разделяются на толстые и тонкие в зависимости от толщины. Толстый коаксиальный кабель имеет большую степень помехозащищенности, большую механическую прочность, но требует специального приспособления для прокалывания кабеля, чтобы создать ответвления для подключения к ЛВС. Он более дорогой и менее гибкий, чем тонкий. Используется в технологии *Ethernet* 10*Base*5 и A*RCNet*.

Оптоволоконный кабель (*Fiber Optic Cable*) обеспечивает высокую скорость передачи данных на большом расстоянии. Они также невосприимчивы к интерференции и подслушиванию. В оптоволоконном кабеле для передачи сигналов используется свет. Волокно, применяемое в качестве световода, позволяет передачу сигналов на большие расстояния с огромной скоростью, но оно дорого, и с ним трудно работать. Для установки разъемов, создания ответвлений, поиска неисправностей в оптоволоконном кабеле необходимы специальные приспособления и высокая квалификация. Оптоволоконный кабель состоит из центральной стеклянной нити толщиной в несколько микрон, покрытой сплошной стеклянной оболочкой. Все это, в свою очередь, спрятано во внешнюю защитную оболочку.

Существуют два типа оптоволоконных кабелей – одномодовые и многомодовые. Одномодовые кабели имеют меньший диаметр, большую стоимость и позволяют передачу информации на большие расстояния.

Методы беспроводной технологии передачи данных (*Radio Waves*) являются удобным, а иногда незаменимым средством связи. Беспроводные технологии различаются по типам сигнала, частоте (большая частота означает большую скорость передачи) и расстоянию передачи. Большое значение имеют помехи и стоимость. Можно выделить три основных типа беспроводной технологии:

- радиосвязь;
- связь в микроволновом диапазоне;
- инфракрасная связь.

Технологии радиосвязи пересылают данные на радиочастотах и практически не имеют ограничений по дальности. Она используется для соединения локальных сетей на больших географических расстояниях. Радиопередача в целом имеет высокую стоимость и чувствительна к электронному и атмосферному наложению, а также подвержена перехватам, поэтому требует шифрования для

обеспечения уровня безопасности.

Передача данных в микроволновом диапазоне (*Microwaves*) использует высокие частоты и применяется как на коротких, так и на больших расстояниях. Главное ограничение заключается в том, чтобы передатчик и приемник были в зоне прямой видимости. Используется в местах, где использование физического носителя затруднено. Передача данных в микроволновом диапазоне при использовании спутников может быть очень дорогой.

Инфракрасные технологии (*Infrared transmission*), функционируют на очень высоких частотах, приближающихся к частотам видимого света. Они могут быть использованы для установления двусторонней или широковещательной передачи на близких расстояниях. При инфракрасной связи обычно используют светодиоды (*LED – Light Emitting Diode*) для передачи инфракрасных волн приемнику. Инфракрасная передача ограничена малым расстоянием в прямой зоне видимости и может быть использована в офисных зданиях.

Важными составляющими сетей являются средства маршрутизации: повторители, мосты, маршрутизаторы, шлюзы. **Повторитель** (*repeater*) – самый простой тип устройства для соединения однотипных ЛВС, он ретранслирует все принимаемые пакеты из одной ЛВС в другую. **Мост** (*bridge*) – устройство связи, позволяющее соединить ЛВС с одинаковыми и разными системами сигналов. **Маршрутизатор** (*router*) – устройство связи, аналогичное мосту, выполняющее функции передачи пакетов в соответствии с определенными протоколами, обеспечивающее соединение ЛВС на сетевом уровне. **Шлюз** (*gateway*) – устройство соединения ЛВС с глобальной сетью. Концентратор (*hub*) – многопортовый повторитель данных из компьютеров, подключенных к этим повторителям, и передающие данные на все свои порты. Он поддерживает несколько каналов передачи данных.

При разработке сети ЭВМ возникает задача согласования взаимодействий ЭВМ клиентов, серверов, линий связи и др. устройств. Информация в сети передается блоками данных по процедурам обмена между объектами. Эти процедуры называют протоколами передачи данных. **Протокол** – это совокупность правил, устанавливающих формат и процедуры обмена информацией между двумя или несколькими устройствами. Часть протоколов реализуется программно, другая часть – аппаратно. Для стандартизации протоколов, т.е. для единого представления данных в сетях с неоднородными устройствами и программным обеспечением, была создана международная организация по стандартизации *ISO* (*International Standardization Organization*), которая разработала базовую модель связи открытых систем *OSI* (*Open System Interconnection*). Эта модель описывает правила и процедуры передачи данных в различных сетевых средах при организации сеанса связи. Основными элементами модели являются

уровни, прикладные процессы и физические средства соединения. Каждый уровень модели *OSI* выполняет определенную задачу в процессе передачи данных по сети. OSI разделяет коммуникационные функции в сети на семь уровней, каждый из которых обслуживает различные части процесса области взаимодействия открытых систем: 1 – физический, 2 – канальный, 3 – сетевой, 4 – транспортный, 5 – сеансовый, 6 – представительный, 7 – прикладной.

Первый уровень (физический) определяет некоторые физические характеристики канала.

Второй уровень (канальный) управляет передачей данных между двумя узлами сети.

Третий уровень (сетевой) обеспечивает управление потоком, маршрутизацию.

Четвертый уровень (транспортный) отвечает за стандартизацию обмена данными между программами, находящимися на разных ЭВМ сети.

Пятый уровень (сеансовый) определяет правила диалога прикладных программ, проверки прав доступа к сетевым ресурсам.

Шестой уровень (представительный) определяет форматы данных, алфавиты, коды представления специальных и графических символов.

Седьмой уровень (прикладной) определяет уровень услуг (электронная почта, телекс, телефакс, видеотекст, телетекст и др.).

Каждый уровень решает свои задачи и обеспечивает сервисом расположенный над ним уровень. Правила взаимодействия разных систем одного уровня называют протоколом, правила взаимодействия соседних уровней в одной системе – интерфейсом. Каждый протокол должен быть прозрачным для соседних уровней, т.е. быть понятным взаимодействующим уровням.

Модель OSI можно разделить на две различных модели:

1) горизонтальную модель на базе протоколов, обеспечивающую механизм взаимодействия программ и процессов на различных машинах;

2) вертикальную модель на основе услуг, обеспечиваемых соседними уровнями друг другу на одной машине.

Каждый уровень компьютера–отправителя взаимодействует с таким же уровнем компьютера-получателя, как будто он связан напрямую. Такая связь называется логической или виртуальной связью. В действительности взаимодействие осуществляется между смежными уровнями одного компьютера.

В горизонтальной модели двум программам требуется общий протокол для обмена данными. В вертикальной модели соседние уровни обмениваются данными с использованием интерфейсов прикладных программ *API* (*Application Programming Interface*).

Перед подачей в сеть данные разбиваются на пакеты. **Пакет** (*packet*) – это единица информации, передаваемая между станциями сети. При отправке дан-

ных пакет проходит последовательно через все уровни программного обеспечения. На каждом уровне к пакету добавляется управляющая информация данного уровня (заголовок), которая необходима для успешной передачи данных по сети. На принимающей стороне пакет проходит через все уровни в обратном порядке. На каждом уровне протокол этого уровня читает информацию пакета, затем удаляет информацию, добавленную к пакету на этом же уровне отправляющей стороной, и передает пакет следующему уровню. Когда пакет дойдет до Прикладного уровня, вся управляющая информация будет удалена из пакета, и данные примут свой первоначальный вид.

Загрузка сети характеризуется параметром, называемым трафиком. **Трафик** (*traffic*) – это поток сообщений в сети передачи данных. Под ним понимают количественное измерение в выбранных точках сети числа проходящих блоков данных и их длины, выраженное в битах в секунду.

Существенное влияние на характеристику сети оказывает метод доступа. **Метод доступа** – это способ определения того, какая из рабочих станций сможет следующей использовать канал связи и как управлять доступом к каналу связи (кабелю).

По способу передачи информации сети принято делить на: коммутации каналов; коммутации сообщений; коммутации пакетов; интегральные сети. Сети коммутации каналов обеспечивают прямое соединение клиентов, которое остается неизменным в течение всего сеанса.

При коммутации сообщений информация передается порциями, называемыми сообщениями. Прямое сообщение не устанавливается. Передача сообщения начинается после освобождения первого канала и так далее, пока сообщение не дойдет до адресата. Каждым сервером осуществляется прием информации, ее сбор.

При коммутации пакетов обмен производится короткими пакетами фиксированной структуры. Пакет – часть сообщения, удовлетворяющая некоторому стандарту. Малая длина пакетов предотвращает блокировку линий связи, не дает расти очереди в узлах коммутации. Это обеспечивает быстрое соединение, низкий уровень ошибок, надежность и эффективность использования сети.

Интегральные сети – сети, обеспечивающие коммутацию каналов, сообщений и пакетов. Они объединяют несколько коммутационных сетей.

Наиболее распространенная услуга сетей – электронная почта. Электронная почта технически – это компьютер, модем и телефон. С другой стороны электронная почта – специальный пакет программ, обеспечивающий хранение и пересылку сообщений между пользователями ЭВМ. Она обеспечивает сбор, регистрацию, обработку и передачу любой информации (текстовых документов, изображений, цифровых данных, звукозаписи и т.д.) по сетям ЭВМ и вы-

полняет функции: редактирования документов перед передачей; их хранения; пересылки корреспонденции; проверки и исправления ошибок, возникающих при передаче; выдачи подтверждения о получении корреспонденции адресатом; получения и хранения информации в своем «почтовом ящике»; просмотра полученной корреспонденции.

По электронной почте можно посылать индивидуальное, групповое или общее сообщение. Электронная почта может быть организована в ЛВС для обеспечения внутреннего объема информации. Программы электронной почты могут применяться автономно и в составе интегрированных систем. Большинство глобальных систем ЭВМ поддерживают электронную почту.

К одному из важных видов сетевых технологий относится распределенная обработка данных. Ее особенность – ПК стоят на рабочих местах, т.е. на местах возникновения и использования информации; они соединены каналами связи, что дает возможность распределять их ресурсы по отдельным функциональным сферам деятельности и изменить технологию обработки данных в направлении децентрализации.

Следует различать распределенную обработку данных и распределенную базу данных. В первом случае база данных находится на сервере, а обработка осуществляется на компьютерах-клиентах. Во втором случае – база данных размещается на нескольких серверах.

В системе распределенной обработки данных клиент работает в режиме запросов. Он может послать запрос к собственной локальной базе или удаленной. Удаленный запрос – единичный запрос к одному серверу. Несколько удаленных запросов к одному серверу объединяются в удаленную транзакцию. Если отдельные запросы транзакций обрабатываются различными серверами, то транзакция называется распределенной. При этом один запрос транзакций обрабатывается одним сервером. Распределенная база данных позволяет обрабатывать один запрос несколькими серверами. Такой запрос называется распределенным.

Различают централизованную, децентрализованную и смешанную технологии распределенной обработки данных. При централизованной обработке данных на одном сервере находится единственная копия базы данных. Все операции с базой данных обеспечиваются этим сервером. Доступ к данным выполняется с помощью удаленного запроса и удаленной транзакции.

Децентрализованная организация данных предполагает разбиение информационной базы на несколько физически распределенных. Каждый клиент пользуется своей базой данных, которая может быть либо частью общей информационной базы, либо копией информационной базы в целом.

В сети все рабочие станции физически соединены между собою каналами

связи по определенной структуре, называемой топологией. **Топология** – это описание физических соединений в сети, указывающее какие рабочие станции могут связываться между собой. Тип топологии определяет производительность, стоимость, защищенность и надежность эксплуатации рабочих станций, а также время обращения к файловому серверу. В зависимости от топологии сети используется тот или иной метод доступа.

Понятие топологии широко используется при создании сетей. Одним из подходов к классификации топологий ЛВС является выделение двух основных классов топологий: широковещательные и последовательные.

В широковещательных топологиях ПК передает сигналы, которые могут быть восприняты остальными ПК. К таким топологиям относятся топологии:

- общая шина (*Bus*);
- звезда (*Star*);
- древовидная (*Tree*).

В последовательных топологиях информация передается только одному ПК. Примерами таких топологий являются:

- кольцо (Ring);
- ячеистая (Mesh);
- произвольная (произвольное соединение ПК);
- цепочка.

При выборе оптимальной топологии преследуются три основных цели:

1) обеспечение альтернативной маршрутизации и максимальной надежности передачи данных;

2) выбор оптимального маршрута передачи блоков данных;

3) предоставление приемлемого времени ответа и нужной пропускной способности.

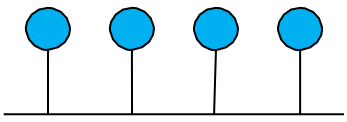


Рис. 60. Пример топологии общая шина

При выборе конкретного типа сети важно учитывать ее топологию.

Топология «**Общая шина**» это тип сетевой топологии, в которой рабочие станции расположены вдоль одного участка кабеля, называемого сегментом (рис. 60).

В случае топологии Общая шина кабель используется всеми станциями по очереди. Принимаются специальные меры для того, чтобы при работе с общим кабелем компьютеры не мешали друг другу передавать и принимать данные. Все сообщения, посылаемые отдельными компьютерами, принимаются и прослушиваются всеми остальными компьютерами, подключенными к сети. Рабочая станция отбирает адресованные ей сообщения, пользуясь адресной информацией. Надежность здесь выше, так как выход из строя отдельных компьютеров не нарушит работоспособность сети в целом. Поиск неисправности в сети затруднен. Кроме того, так как используется только один кабель, в случае обрыва нарушается работа всей сети.

Шинная топология - это наиболее простая и наиболее распространенная топология сети.

Топология «**Кольцо**» – это топология ЛВС, в которой каждая станция соединена с двумя другими станциями, образуя кольцо (рис. 61). Данные передаются от одной рабочей станции к другой в одном направлении (по кольцу). Каждый ПК работает как повторитель, ретранслируя сообщения к следующему ПК, т.е. данные, передаются от одного компьютера к другому как бы по эстафете. Если компьютер получает данные, предназначенные для другого компьютера, он передает их дальше по кольцу, в ином случае они дальше не передаются. Очень просто делается запрос на все станции одновременно. Основная проблема при кольцевой топологии заключается в том, что каждая рабочая станция должна активно участвовать в пересылке информации, и в случае выхода из строя хотя бы одной из них, вся сеть парализуется. Подключение новой рабочей станции требует краткосрочного выключения сети, т.к. во время установки кольцо должно быть разомкнуто. Топология Кольцо имеет хорошо предсказуемое время отклика, определяемое числом рабочих станций. Чистая кольцевая топология используется редко. Вместо этого кольцевая топология играет транспортную роль в схеме метода доступа. Кольцо описывает логический маршрут, а пакет передается от одной станции к другой, совершая в итоге полный круг.

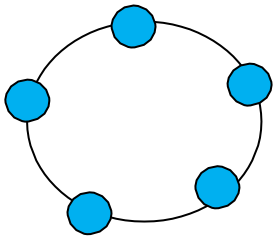


Рис. 61. Пример топологии кольцо

Топология «**Звезда**» – это топология ЛВС, в которой все рабочие станции присоединены к центральному узлу (например, к концентратору), который устанавливает, поддерживает и разрывает связи между рабочими станциями (рис. 62). Преимуществом такой топологии является возможность простого исключения неисправного узла. Однако, если неисправен центральный узел, вся сеть выходит из строя.

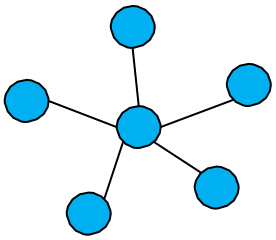


Рис. 62. Пример топологии звезда

В этом случае каждый компьютер через специальный сетевой адаптер подключается отдельным кабелем к объединяющему устройству. При необходимости можно объединять вместе несколько сетей с топологией Звезда, при этом получаются разветвленные конфигурации сети. В каждой точке ветвления необходимо использовать специальные соединители (распределители, повторители или устройства доступа).

Звездообразная топология обеспечивает защиту от разрыва кабеля. Если кабель рабочей станции будет поврежден, это не приведет к выходу из строя всего сегмента сети. Она позволяет также легко диагностировать проблемы подключения, так как каждая рабочая станция имеет свой собственный кабельный сегмент, подключенный к концентратору. Для диагностики достаточно найти разрыв кабеля, который ведет к неработающей станции. Остальная часть сети продолжает нормально работать.

Однако звездообразная топология имеет и недостатки: требует много кабеля, концентраторы довольно дороги, кабельные концентраторы при большом количестве кабеля трудно обслуживать.

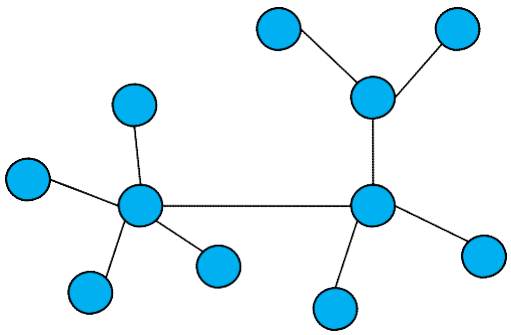


Рис. 63. Пример топологии дерево

Топология «**Дерево**» представляет собой структуру, аналогичную линейной, с той лишь разницей, что здесь возможны ветви с несколькими узлами (рис. 63).

Данную топологию можно рассматривать как комбинацию нескольких звезд. Как и в случае звезды, дерево может быть активным, или настоящим, и пассивным. При активном дереве в центрах объединения нескольких линий связи находятся централь-

ные компьютеры, а при пассивном – концентраторы (хабы). В такой сети между любыми двумя узлами существует только один путь.

В этом случае каждый компьютер через специальный сетевой адаптер подключается отдельным кабелем к объединяющему устройству. При необходимости можно объединять вместе несколько сетей с топологией Звезда, при этом получаются разветвленные конфигурации сети. В каждой точке ветвления необходимо использовать специальные соединители (распределители, повторители или устройства доступа).

Звездообразная топология обеспечивает защиту от разрыва кабеля. Если кабель рабочей станции будет поврежден, это не приведет к выходу из строя всего сегмента сети. Она позволяет также легко диагностировать проблемы подключения, так как каждая рабочая станция имеет свой собственный кабельный сегмент, подключенный к концентратору. Для диагностики достаточно найти разрыв кабеля, который ведет к неработающей станции. Остальная часть сети продолжает нормально работать.

Однако звездообразная топология имеет и недостатки: требует много кабеля, концентраторы довольно дороги, кабельные концентраторы при большом количестве кабеля трудно обслуживать.

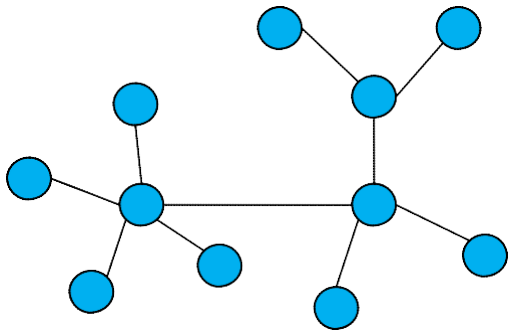


Рис. 63. Пример топологии дерево

Топология **«Дерево»** представляет собой структуру, аналогичную линейной, с той лишь разницей, что здесь возможны ветви с несколькими узлами (рис. 63).

Данную топологию можно рассматривать как комбинацию нескольких звезд. Как и в случае звезды, дерево может быть активным, или настоящим, и пассивным. При активном дереве в центрах объединения нескольких линий связи находятся центральные компьютеры, а при пассивном – концентраторы (хабы). В такой сети между любыми двумя узлами существует только один путь.

Вопрос 2. Интернет-технология

Интернет – глобальная компьютерная сеть, связывающая между собой как пользователей компьютерных сетей, так и пользователей индивидуальных (в том числе домашних) компьютеров различного назначения, выполняющих разные задачи.

В 1957 году после запуска Советским Союзом искусственного спутника Земли Министерство обороны США посчитало, что на случай войны Америке нужна надежная система передачи информации. Агентство передовых оборонных исследовательских проектов США (*DARPA*) предложило разработать для этого компьютерную сеть. Компьютерная сеть была названа *ARPANET* (англ. *Advanced Research Projects Agency Network*). Затем сеть *ARPANET* начала активно расти и развиваться, ее начали использовать ученые из разных областей науки.

Первый сервер *ARPANET* был установлен в 1969 году в Калифорнийском университете в Лос-Анджелесе и в этом же году между двумя первыми узлами сети *ARPANET,* находящимися на расстоянии в 640 км провели первый сеанс связи. Именно это событие можно считать днем рождения Интернета.

К 1971 году была разработана первая программа для отправки электронной почты по сети. Эта программа сразу стала очень популярна. В 1973 году к сети были подключены через трансатлантический телефонный кабель первые иностранные организации из Великобритании и Норвегии, сеть стала международной.

В 1970-х годах сеть в основном использовалась для пересылки электронной почты, тогда же появились первые списки почтовой рассылки, новостные группы и доски объявлений. Однако в то время сеть еще не могла легко взаимодействовать с другими сетями, построенными на других технических стандартах. К концу 1970-х годов начали бурно развиваться протоколы передачи данных, которые были стандартизированы в 1982–1983 годах. Активную роль в разработке и стандартизации сетевых протоколов играл Джон Постел. 1 января 1983 года сеть *ARPANET* перешла с протокола *NCP* на *TCP/IP*, который успешно применяется до сих пор для объединения сетей. Именно в 1983 году термин «Интернет» закрепился за сетью *ARPANET*.

В 1984 году была разработана система доменных имен (англ. *Domain Name System, DNS*). В 1984 году у сети *ARPANET* появился серьезный соперник: Национальный научный фонд США (*NSF*) основал обширную межуниверситетскую сеть *NSFNet* (англ. *National Science Foundation Network*), которая была составлена из более мелких сетей (включая известные тогда сети *Usenet* и *Bitnet*) и имела гораздо большую пропускную способность, чем *ARPANET*. К этой сети за год подключились около 10 тыс. компьютеров, звание «Интернет» начало

плавно переходить к *NSFNet*.

В 1988 году был разработан протокол *Internet Relay Chat* (*IRC*), благодаря чему в Интернете стало возможно общение в реальном времени (чат).

В 1989 году в Европе, в стенах Европейского совета по ядерным исследованиям (фр. *Conseil Européen pour la Recherche Nucléaire,CERN*) родилась концепция Всемирной паутины. Ее предложил знаменитый британский ученый Тим Бернерс-Ли, он же в течение двух лет разработал протокол *HTTP*, язык *HTML* и текстовые идентификаторы *URI*.

В 1990 году сеть *ARPANET* прекратила свое существование, полностью проиграв конкуренцию *NSFNet*. В том же году было зафиксировано первое подключение к Интернету по телефонной линии.

В 1991 году Всемирная паутина стала общедоступна в Интернете, а в 1993 году появился знаменитый веб-браузер *NCSAMosaic*. Всемирная паутина набирала популярность

В 1995 году *NSFNet* вернулась к роли исследовательской сети, маршрутизацией всего трафика Интернета теперь занимались сетевые провайдеры, а не суперкомпьютеры Национального научного фонда. В том же 1995 году Всемирная паутина стала основным поставщиком информации в Интернете, обогнав по трафику протокол пересылки файлов *FTP*. Был образован Консорциум всемирной паутины (*W3C*). Можно сказать, что Всемирная паутина преобразила Интернет и создала его современный облик. С 1996 года Всемирная паутина почти полностью подменяет собой понятие «Интернет».

В 1990-е годы Интернет объединил в себе большинство существовавших тогда сетей (хотя некоторые, как Фидонет, остались обособленными). Объединение выглядело привлекательным благодаря отсутствию единого руководства, а также благодаря открытости технических стандартов Интернета, что делало сети независимыми от бизнеса и конкретных компаний. К 1997 году в Интернете насчитывалось уже около 10 миллионов компьютеров, было зарегистрировано более 1 миллиона доменных имен. Интернет стал очень популярным средством для обмена информацией.

В начале февраля 2011 произошло знаковое событие в истории глобальной сети. Организация *ICANN* (*Internet Corporation for Assigned Names and Numbers*) выделила из центрального пула последние блоки адресов *IPv4*. Это означает, что дельнейшее расширение Интернет зависит только от успешности перехода на новое поколение интернет-протокола, *IPv6*.

В настоящее время подключиться к Интернету можно через спутники связи, радио-каналы, кабельное телевидение, телефон, сотовую связь, специальные оптико-волоконные линии или электропровода. Всемирная сеть стала неотъемлемой частью жизни в развитых и развивающихся странах.

Рунет – русскоязычная часть всемирной сети Интернет. Или можно сказать, что Рунет – это часть Всемирной паутины, принадлежащая к национальным доменам *.su*, *.ru* и .рф. В 1990 году профессиональная научная сеть, выросшая в недрах Института атомной энергии им. И. В. Курчатова и ИПК Минавтопрома и объединившая ученых-физиков и программистов, соединилась с мировой сетью Интернет, положив начало современным российским сетям. 19 сентября 1990 года был зарегистрирован домен первого уровня *.su* в базе данных Международного информационного центра *InterNIC*. В результате этого Советский Союз стал доступен через Интернет. 7 апреля 1994 года в InterNIC был зарегистрирован российский домен .ru. Домен.рф, позволяющий использовать в адресе URL кириллические символы, делегирован в корневой зоне DNS в 2010 году.

Каждый компьютер в Интернете имеет свой уникальный адрес, который состоит из четырех чисел, находящихся в диапазоне от 0 до 255 и разделенных точками. Вот пример такого адреса:192.254.55.232.

Такие адреса называются IP-адресами, потому что они обеспечивают корректную работу протокола *IP*.

Описываемая система адресации основана на *IP* версии 4, использующей 32-битовую адресацию. Каждое из четырех чисел адреса соответствует восьми битам информации. Поэтому эти числа называются октетами. Такое адресное пространство позволяет использовать 232 (примерно 4,3 миллиарда) различных адресов. Когда Винтон Серф и его коллеги занимались разработкой протокола *TCP/IP*, они, конечно, не могли представить себе темпов роста глобальной сети. В настоящее время Интернет столкнулся с реальной угрозой нехватки адресов. Поэтому не так давно была разработана версия *IP* 6 (называемая также «*IPng*», или «*IP Next Generation*»), использующая 128-битную адресацию.

IP-адрес состоит из двух частей. Первая – к ней относятся от одного до трех чисел слева – обозначает сеть, в которой находится компьютер, и называется идентификатором сети. Интернет состоит из множества сетей, каждая из которых имеет собственный адрес. Вторая часть *IP*-адреса – соответственно, от одного до трех чисел справа – обозначает конкретный компьютер в сети и называется идентификатором узла. Таким образом, иерархия *IP*-адресов читается слева направо, то есть слева располагаются старшие биты, справа – младшие.Очевидно, чем больше первое число в *IP*-адресе (то есть чем больше в нем битов), тем большее количество адресов можно образовать с его использованием. Поэтому такие числа присутствуют в адресах больших сетей. Наоборот, меньшие идентификаторы сети говорят о меньшем размере сети.

В зависимости от своего размера сети поделены на четыре класса – *A*, *B*, *C*, и *D*, где *A* обозначает самые большие сети, *D* – самые маленькие.

Первый октет *IP*-адреса сети класса *A* находится в диапазоне от 1 до 126. Количество узлов в такой сети может достигать 16777214. Идентификаторы распределяются в адресе следующим образом: сеть.узел.узел.узел.

Первый октет *IP*-адреса сети класса *C* находится в диапазоне от 192 до 233. Количество узлов в этом случае ограничено 254. Адрес каждого узла выглядит так: сеть. сеть, сеть. узел.

Присвоением адресов в Интернете занимается организация под названием *InterNIC* (*Network Information Center*). При регистрации сети в Интернете ей выделяется сетевой идентификатор в зависимости от ее класса. Идентификацию узлов в пределах сети выполняет организация-владелец.

Многие организации, имеющие в своем распоряжении большие сети (например, провайдеры услуг Интернета), иногда «экономят» на *IP*-адресах. Они резервируют меньшее их количество, чем число узлов в сети. В этом случае каждому узлу при подключении выделяется динамический *IP*-адрес из тех, которые свободны в данный момент. Когда компьютер подключается к Интернету, он становится его частью, а значит, ему должен быть присвоен уникальный *IP*-адрес. Он получает его при каждом подключении, но этот адрес каждый раз имеет новое значение.

Интернет-провайдер (*InternetServiceProvider, ISP*) – компания, предоставляющая услуги по подключению к сети Интернет.

Статические *IP*-адреса, как правило, закреплены за теми узлами Интернета, которые должны присутствовать в Сети постоянно. Это серверы, назначение которых состоит в том, чтобы обрабатывать запросы пользователей Интернета.

Для более удобного человеку представления адресов Интернета, отличного от представления *IP*-адреса,была разработана система доменных имен. Применительно к Интернету домен является виртуальной зоной, к которой относится тот или иной компьютер.

Доменное имя представляет адрес любого ресурса в Интернете как последовательность слов. Некоторые из них имеют смысловое содержание и легко запоминаются. Адреса Интернет-ресурсов, представленные таким образом, называют *URL-Uniform Resource Locator*, универсальный указатель ресурса.

Вот типичные примеры доменных имен:*www.lenta.ru, www.yahoo.com.*

В отличие от *IP*-адресов иерархия доменных имен читается справа налево. Самый правый сегмент доменного имени представляет собой домен верхнего уровня. В настоящее время Интернет поделен на домены верхнего уровня по одному из двух признаков: либо по географическому, либо по признаку характера деятельности.

Домены верхнего уровня, различающиеся по признаку деятельности сайта:

- com – коммерческие организации;

- edu – образовательные учреждения;
- gov – правительственные организации;
- mil – военные организации;
- net – организации, имеющие, как правило, отношение к сетевым ресурсам;
- org – общественныеорганизации.

Поскольку для человека предпочтительны доменные имена, а для компьютера – *IP*-адреса, между этими двумя вариантами записи адреса установлены однозначные соответствия. Когда компьютеру дается команда открыть страницу через определенный *URL*, то он обращается за справкой к другому компьютеру, чтобы определить, какой IP-адрес скрывается за введенным доменным именем. Этот «справочный» компьютер называется сервером DNS. Domain Name System – служба каталогизации доменных имен. Таблица соответствия доменных имен IP-адресам размещается на многих DNS-серверах, которые последовательно опрашиваются при поиске того или иного значения.

В России регистрацией доменных имен в домене *ru* занимается организация под названием РосНИИРОС (Российский НИИ Развития Общественных сетей, *www.ripn.ru*). Эта процедура платная.

Простое подключение одного компьютера к другому – шаг, необходимый для создания сети, но недостаточный. Для обмена информацией необходимо, чтобы компьютеры «понимали» друг друга. Для этого были разработаны специальные средства, получившие название протоколов.

Протокол – это совокупность правил, в соответствии с которыми происходит передача информации через сеть.

Над созданием протоколов, необходимых для существования глобальной сети, трудились различные ученые. Одним из них был Винтон Серф, которого сейчас называют «отцом Интернета». В 1972 году группа разработчиков под руководством Винтона Серфа разработала протокол *TCP/IP-Transmission Control Protocol/Internet Protocol* (Протокол управления передачей/Протокол Интернета).

Эксперимент по разработке этого протокола проводился по заказу Министерства обороны США. Поэтому главной задачей при разработке сетевого протокола являлась его «неприхотливость» – он должен был работать с любым сетевым окружением и, кроме того, обладать гибкостью в выборе маршрута при доставке информации.

Позже *TCP/IP* перерос свое изначальное предназначение и стал основой стремительно развивавшейся глобальной сети, ныне известной как Интернет, а также небольших сетей, использующих технологии Интернета, – интранет. Стандарты *TCP/IP* являются открытыми и непрерывно совершенствуются.

На самом деле *TCP/IP* является не одним протоколом, а целым набором протоколов, работающих совместно. Он состоит из двух уровней. Протокол верхнего уровня, *TCP*, отвечает за правильность преобразования сообщений в пакеты информации, из которых на приемной стороне собирается исходное послание. Протокол нижнего уровня, *IP*, отвечает за правильность доставки сообщений по указанному адресу – иногда пакеты одного сообщения могут доставляться разными путями.

Протокол *HTTP* (*Hypertext Transfer Protocol* – Протокол передачи гипертекста) является протоколом более высокого уровня по отношению к протоколу *TCP/IP* – протоколом уровня приложения. *HTTP* был разработан для эффективной передачи по Интернету *Web*-страниц. Протокол *HTTP* является основой системы *World Wide Web*.

Команды *HTTP* отдаются с использованием интерфейса браузера, который является *HTTP*-клиентом. При щелчке мышью на ссылке браузер запрашивает у *Web*-сервера данные того ресурса, на который указывает ссылка, например, очередной *Web*-страницы. Адреса ресурсов Интернета, к которым происходит обращение по протоколу *HTTP*, выглядят примерно следующим образом: *http://www.utmn.ru*

Протокол *FTP* (*File Transfer Protocol* – Протокол передачи файлов) специально разработан для передачи файлов по Интернету. Назначение электронной почты – прежде всего обмен текстовой информацией между различными компьютерными системами. Не меньший интерес для пользователей сети Интернет представляет обмен отдельными файлами и целыми программами.

Для того, чтобы обеспечить перемещение данных между различными операционными системами, которые могут встретиться в Интернет, используется протокол *FTP*, работающий независимо от применяемого оборудования. Пользователь получает доступ к различным файлам и программам, хранящимся на компьютерах, подключенных к сети.

Программа, реализующая этот протокол, позволяет установить связь с одним из множества *FTP*-серверов в Интернет. *FTP*-сервер – компьютер, на котором содержатся файлы, предназначенные для открытого доступа. Программа *FTP*-клиент не только реализует протокол передачи данных, но и поддерживает набор команд, которые используются для просмотра каталога *FTP*-сервера, поиска файлов и управления перемещением данных.

С помощью протокола *Telnet* можно подключиться к удаленному компьютеру как пользователь (если известно имя пользователя и пароль) и производить действия над его файлами и приложениями точно так же, как если бы работа выполнялась на своем компьютере. *Telnet* является протоколом эмуляции терминала. Работа с ним ведется из командной строки.

WAIS расшифровывается как *Wide-Area Information Servers*. Этот протокол был разработан для поиска информации в базах данных. Информационная система *WAIS* представляет собой распределенную базу данных, где отдельные базы данных хранятся на разных серверах. Сведения об их содержании и расположении хранятся в специальной базе данных – каталоге серверов. Просмотр информационных ресурсов осуществляется с помощью программы-клиента *WAIS*.

Поиск информации ведется по ключевым словам, которые задает пользователь. Эти слова вводятся для определенной базы данных, и система находит все соответствующие им фрагменты текста на всех серверах, где располагаются данные этой базы. Результат представляется в виде списка ссылок на документы с указанием того, насколько часто встречается в данном документе искомое слово и все искомые слова в совокупности.

Протокол Gopher – протокол уровня приложения. До повсеместного распространения гипертекстовой системы World Wide Web Gopher использовался для извлечения информации (в основном текстовой) из иерархической файловой структуры. Gopher позволял с помощью меню передвигаться от одной страницы к другой, постепенно сужая круг отображаемой информации. Программы-клиенты Gopher имели текстовый интерфейс. Однако пункты меню *Gopher* могли указывать и не только на текстовые файлы, но также, например, на *telnet*-соединення или базы данных *WAIS*. Сейчас ресурсы *Gopher* можно просматривать с помощью обычного *Web*-браузера, так как современные браузеры поддерживают этот протокол.

Для подключения пользователя к *Internet* используются разнообразные технологии, которые имеют следующие характеристики:

- Вид передачи информации: беспроводной, через телефонный, телевизионный или сетевой провод.
- Стоимость: за беспроводные подключения приходится платить больше, но проводные имеют высокую цену «за проводку и подключение».
- Скорость: для комфортного использования Интернета достаточно 1-3 мегабит/с, но при условии стабильного подключения. Скорее влияет на скорость загрузки больших файлов, чем на скорость в целом.
- Стабильность: частота разрывов связи.
- Время отклика: пинг (скорость ответа от сервера). Определяет скорость загрузки множества маленьких файлов. Сильно влияет на скорость открытия страниц.

Рассмотрим некоторые конкретные виды подключений:

1. Спутниковый Интернет имеет самый широкий радиус покрытия: почти

на любой точке земного шара им можно воспользоваться. Однако у него есть ряд недостатков: высокая стоимость, дорогое оборудование, зависимость от погодных условий, низкая скорость отклика.

2. Мобильный Интернет, к которому относятся подключения через мобильный телефон либо беспроводной модем. Относительно старые телефоны подключаются по медленной и дорогой технологии *GPRS*, а новые телефоны используют более скоростные: *CDMA*, *UMTS*, *LTE*, *WiMAX*. Для последних технологий операторы связи почти всегда предлагают модем, который можно использовать в компьютере или ноутбуке и без проблем пользоваться Интернетом везде, где есть телефонное покрытие.

3. *Wi-Fi* (англ. *Wireless Fidelity*) – это отдельный вид беспроводного подключения для беспроводных сетей на базе стандарта IEEE 802.11. Сейчас почти в каждом более-менее современном телефоне и ноутбуке есть приемник для этой технологии. Подключение производится к «точкам доступа» с радиусом действия до 100 метров. Провайдеров для этой технологии очень мало, т.к. чаще всего эту технологии используют для частной раздачи Интернета в публичных местах, кафе, аэропортах как дополнительную бесплатную услугу.

4. Коммутируемый удаленный доступ (англ. *dial-up*) – сервис, позволяющий компьютеру, используя модем и телефонную сеть общего пользования, подключаться к Интернету.

5. Прямое подключение. В этом случае от провайдера к пользователю идет обычный сетевой компьютерный кабель. Лучший вариант подключения в настоящий момент. Обладает наилучшей стабильностью и более низким пингом в сравнении с другими видами подключения, также на аналогичных скоростях доступа имеет более низкую стоимость.

Для подключения корпоративной или локальной сети к Интернет используется *proxy*-сервер, осуществляющий фильтрацию пакетов, передаваемых между Интернет и локальной сетью. Основное назначение *proxy*-сервера – кэширование, т.е. сохранение наиболее часто запрашиваемых документов, что существенно снижает сетевой трафик от провайдера к локальной сети. Также он может использоваться для установки настроек и ограничений на работу каждого пользователя Интернет из локальной сети. Отчасти *proxy*-сервер повышает безопасность локальных сетей, так как затрудняет несанкционированное проникновение в сеть извне.

К информационно-коммуникационным *off-line* сервисам сетиИнтернет можно отнести электронную почту (*E-mail*) и сервисы, функционирующие на ее базе: телеконференции, списки рассылок, доски объявлений и др. Коммуникационные сервисы работающие в режиме реального времени, – *on-line* сервисы,

или сервисы и программы для прямого межпользовательского общения: *Talk*, *IRC* (*InternetRelayChat*), *Chat-rooms* (виртуальные гостиные), *ICQ* (*Iseekyou*), *NetMeeting* (обмен файлами, сообщениями), службы *Internet-Phone* (обмен голосовыми сообщениями).

Телеконференции являются сетевым сервисом, ориентированным на поддержку коллективных дискуссий, в которых могут принимать участие тысячи пользователей глобальных компьютерных сетей, и основная цель телеконференций – предоставление оперативной информации.

Телеконференция – это возможность пользователя информационной компьютерной сети участвовать вместе с другими пользователями в обсуждении определенной темы. Телеконференцию в классическом понимании можно представить как большую доску объявлений. Абонент, подписанный на определенную телеконференцию, получает статьи по выбранной теме, написанные другими абонентами. В этом смысле телеконференция напоминает периодическое тематическое издание. Однако абонент может послать в конференцию и свою статью, которая станет доступной всем участникам конференции.

Internet Relay Chat представляет из себя систему, которая позволяет вести диалог с другими пользователями Интернет. IRC – многопользовательская система общения, в которой люди общаются на специальных «каналах» или лично. Каналы (channels) можно сравнить с комнатами, если пользователь заходит на канал, то любая фраза может быть услышана всеми, кто находится на том же канале – вне зависимости от того в какой стране они находятся. При необходимости можно общаться лично, в этом случае сообщение увидит только тот, кому оно предназначено.

ICQ – это *Internet*-технология, позволяющая в конкретный (данный) момент времени видеть, кто из друзей (знакомых, партнеров, клиентов) подключен к Интернет, и обеспечивающая различные возможности общения с ними. *ICQ* находит друзей и в режиме реального времени предупреждает, когда они будут на линии. Отпадает нужда каждый раз выполнять поиск машины в локальной сети, чтобы передать что-нибудь.

С помощью *ICQ* можно беседовать, оставлять сообщения, передавать файлы и *URL*, играть в сетевые игры или просто общаться с людьми во время хождения по сети. *ICQ* может быть использована в многопользовательском режиме, так что группы людей и компании могут с ее помощью проводить живые конференции.

Поисковая машина – поисковая система с формируемой программным «роботом» базой данных, содержащей информацию об информационных ресурсах.

Главной задачей любой поисковой машина является поиск информации, соответствующей информационным потребностям пользователя. Поиск в такой системе проводится по запросу, составляемому пользователем, состоящему из набора ключевых слов или фразы. Многие поисковые системы позволяют проводить поиск в найденных документах, причем можно уточнить запрос введением дополнительных терминов. Некоторые поисковые машины позволяют провести пересортировку результатов.

Наиболее известные и популярные системы для поиска информации: Апорт – www.aport.ru, *Яndex*–www.yandex.ru, *Google* –www.google.ru, *Rambler* – www.rambler.ru, *Yahoo*! – www.yahoo.com, *AltaVista* – www.altavista.com, *InfoSeek* – www.infoseek.com.

Кроме рассмотренных, существуют также системы для поиска файлов (*files.ru*), людей (*whowhere.ru*) и т. д.

Для вызова поисковой системы, пользователю необходимо ввести в адресной строке браузераее адрес (например, www.yandex.ru или www.ya.ru). После загрузки поисковой системы в строке для поиска вводится запрос (ключевая фраза), который представляет собой строку текста (на русском, английском или любом другом языке), и нажимается кнопка Найти.

Для лучшего поиска информации можно использовать следующие правила:

1) поиск осуществлять больше, чем по одному ключевому слову;

2) проверять орфографию;

3) использовать синонимы, если список найденных страниц слишком мал или не содержит полезных страниц;

4) если один из найденных документов ближе к искомой теме, чем остальные, использовать ссылку «найти похожие документы» – ссылка расположена под краткими описаниями найденных документов;

5) чтобы исключить документы, где встречается определенное слово, использовать знак «–», а чтобы определенное слово обязательно присутствовало в документе, поставить перед ним «+»;

6) для того, чтобы воспользоваться широким спектром возможностей, работать со страницей Расширенный поиск, где можно указать искомые слова, язык, дату обновления и формат документа.

Вопрос 3. Гипертекстовая и мультимедиа технологии

Суть **гипертекстовой технологии** заключается в том, что текст представляется как многомерный, т.е. с иерархической структурой типа сети. Материал текста делится на фрагменты. Каждый видимый на экране ЭВМ фрагмент, до-

полненный многочисленными связями с другими фрагментами, позволяет уточнить информацию об изучаемом объекте и двигаться в одном или нескольких направлениях по выбранной связи.

Под гипертекстом понимают систему информационных объектов (статей), объединенных между собой направленными связями, образующими сеть. Каждый объект связывается с информационной панелью экрана, на которой пользователь может выбирать одну из связей. Объекты могут быть текстовыми, графическими, музыкальными, с использованием средств мультипликации, аудио- и видеотехники. Вместо традиционных методов поиска информации по соответствующему поисковому ключу гипертекстовая технология предполагает перемещение от одних объектов информации к другим с учетом их смысловой, семантической связанности.

Структурно гипертекст состоит из информационного материала, тезауруса гипертекста, списка главных тем и алфавитного словаря.

Информационный материал подразделяется на информационные статьи, состоящие из заголовка статьи и текста. Заголовок содержит тему или наименование описываемого объекта. Информационная статья содержит традиционные определения и понятия, должна занимать одну панель и быть легко обозримой.

Тезаурус гипертекста – это автоматизированный словарь, отображающий семантические отношения между лексическими единицами дескрипторного информационно-поискового языка и предназначенный для поиска слов по их смысловому содержанию. Он состоит из тезаурусных статей, которые имеют заголовки и список заголовков родственных тезаурусных статей. Заголовок тезаурусной статьи совпадает с наименованием информационной статьи и является наименованием объекта, описание которого содержится в информационной статье.

Список главных тем содержит заголовки всех справочных статей, для которых нет ссылок типа род – вид, часть – целое.

Алфавитный словарь включает в себя перечень наименований всех информационных статей в алфавитном порядке.

Гипертекстовая технология применяется в издательской деятельности, библиотечной работе, обучающих системах, при разработке документации, законов, справочных руководств, баз данных, баз знаний и т. д.

Мультимедиа – интерактивная технология, обеспечивающая работу с неподвижными изображениями, видеоизображением, анимацией, текстом и аудиоинформацией. Одним из первых инструментальных средств создания технологии мультимедиа явилась гипертекстовая технология, которая обеспечивает работу с текстовой информацией, изображением, звуком, речью. Появлению

систем мультимедиа способствовал технический прогресс: увеличение оперативной и внешней памяти ЭВМ, появление широких графических возможностей ЭВМ, повышение качества видеотехники; возникновение лазерных компакт-дисков и др. Теле-, видео- и аудиоаппаратура в отличие от компьютеров имеет дело с аналоговым сигналом. Поэтому встала проблема стыковки разнородной аппаратуры с компьютером и управления им. Были разработаны звуковые платы (*SoundBlaster*), платы мультимедиа, которые аппаратно реализуют алгоритм перевода аналогового сигнала в дискретный. К компакт-дискам было подсоединено постоянное запоминающее устройство. В 1988 г. был создан принципиально новый тип персонального компьютера – *NeXT*, у которого базовые средства систем мультимедиа заложены в архитектуру, аппаратные и программные средства.

Технологию мультимедиа реализуют специальные программные средства, имеющие встроенную поддержку мультимедиа и позволяющие использовать ее в профессиональной деятельности, учебно-образовательных, научно-популярных и игровых областях.

При применении этой технологии в экономической работе открываются реальные перспективы использовать компьютер для озвучивания изображений, а также понимания им человеческой речи, ведения компьютером диалога со специалистом на родном для специалиста языке. Способность компьютера с голоса воспринимать несложные команды управления программами, открытием файлов, выводом информации на печать и другими операциями в ближайшем будущем создаст благоприятные условия пользователю для взаимодействия с ним в процессе профессиональной деятельности.

ТЕМА 9. ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

План:

1. Классификация угроз информационной безопасности
2. Меры противодействия угрозам
3. Вредительские программы
4. Программные средства защиты информации

С развитием вычислительных средств и систем передачи информации все более актуальной становится проблема обеспечения ее безопасности, предотвращения несанкционированного доступа к информации, физического уничтожения или модификации защищаемой информации.

Вопрос 1. Классификация угроз информационной безопасности

Под **угрозами безопасности информации** (безопасности информационных систем) понимаются потенциально возможные события, процессы или явления, которые могут привести к уничтожению, утрате целостности, конфиденциальности или доступности информации.

Под угрозой безопасности информации понимается действие или событие, которое может привести к разрушению, искажению или несанкционированному использованию информационных ресурсов.

Классификацию источников угроз и их проявлений, целесообразно проводить на основе анализа взаимодействия логической цепочки: источник угрозы – фактор (уязвимость) – угроза (действие) – последствия (атака).

Под этими терминами будем понимать:

Источник угрозы – это потенциальные антропогенные, техногенные или стихийные носители угрозы безопасности.

Угроза (**действие**) – это возможная опасность (потенциальная или реально существующая) совершения какого-либо деяния (действия или бездействия), направленного против объекта защиты (информационных ресурсов), наносящего ущерб собственнику, владельцу или пользователю, проявляющегося в опасности искажения и потери информации.

Фактор (**уязвимость**) – это присущие объекту информатизации причины, приводящие к нарушению безопасности информации на конкретном объекте и обусловленные недостатками процесса функционирования объекта информатизации, свойствами архитектуры автоматизированной системы, протоколами обмена и интерфейсами, применяемыми программным обеспечением и аппаратной платформой, условиями эксплуатации.

Последствия (**атака**) – это возможные последствия реализации угрозы (возможные действия) при взаимодействии источника угрозы через имеющиеся факторы (уязвимости).

Как видно из определения, атака – это всегда пара «источник – фактор», реализующая угрозу и приводящая к ущербу. При этом, анализ последствий предполагает проведение анализа возможного ущерба и выбора методов парирования угроз безопасности информации.

Существует множество подходов к классификации угроз безопасности информации.

1. Угрозы по природе происхождения делятся на:

- естественные – природные (стихийные бедствия, магнитные бури, радиоактивное излучение, осадки) и технические (связаны надежностью технических средств, обработки информации и систем обеспечения);

- искусственные – случайные (совершенные по незнанию и без злого умысла, из любопытности или халатности, например, отказы и сбои аппаратуры, помехи на линиях связи от внешних воздействий, ошибки человека как звена информационной системы и т.д.) и преднамеренные (вызванные или связанные с действиями человека, например, традиционный шпионаж и диверсии, несанкционированный доступ к информации, электромагнитные излучения и наводки, несанкционированная модификация структур информационной системы и т.д.).

2. Угрозы по направлению осуществления:

- внешние угрозы – исходят от источников, не входящих в состав информационной системы;

- внутренние угрозы – порождаются внутри самой информационной системы.

3. Угрозы по способу осуществления:

- информационные угрозы – воздействие на саму информацию, БД через нарушение технологий обмена и обработки информации;

- программно-аппаратные – воздействие на ПО;

- физические – угрозы, связанные с физическим воздействием на аппаратно- технические элементы компьютерной системы, персонал и по- мещения;

- радиоэлектронные – через э/м каналы, линии связи и питания (внутренние и внешние);

- организационно-правовые – через недоработки в организационных схемах и нормативно-правовой документации систем защиты информации.

4. Угрозы по этапу жизненного цикла компьютерной системы:

- на этапе разработкии производства аппаратно-программных средств;

- на этапе проектированияи ввода в эксплуатацию;
- на этапе эксплуатации компьютерной системы;
- весь спектр угроз на этапе вывода из эксплуатации: носители, алгоритмы, документация и т.п.

К основным угрозам безопасности информации относятся:

1) раскрытие конфиденциальной информации– средствами реализации угрозы раскрытия конфиденциальной информации могут быть несанкционированный доступ к базам данных, прослушивание каналов и т.п;

2) компрометация информации, как правило, реализуется посредством внесения несанкционированных изменений в базы данных, в результате чего ее пользователь вынужден либо отказаться от нее, либо предпринимать дополнительные усилия для выявления изменений и восстановления истинных сведений;

3) несанкционированное использование информационных ресурсов, с одной стороны, является средством раскрытия или компрометации информации, а с другой – имеет самостоятельное значение, поскольку, даже не касаясь пользовательской или системной информации, может нанести определенный ущерб абонентам и администрации;

4) ошибочное использование информационных ресурсовчаще всего является следствием ошибок, имеющихся в программном обеспечении;

5) несанкционированный обмен информациеймежду абонентами может привести к получению одним из них сведений, доступ к которым ему запрещен;

6) отказ от информациисостоит в непризнании получателем или отправителем этой информации фактов ее получения или отправки;

7) отказ в обслуживаниипредставляет собой весьма существенную и распространенную угрозу, источником которой является сами АИС (АИТ).

Наиболее распространенными путями несанкционированного доступа к информации являются:

- перехват электронных излучений;
- принудительное электромагнитное облучение линий связи с целью получения паразитной модуляции несущей;
- применение подслушивающих устройств;
- дистанционное фотографирование;
- перехват акустических излучений и восстановление текста принтера;
- хищение носителей информации и документальных отходов;
- чтение остаточной информации в памяти системы после выполнения санкционированных запросов;
- копирование носителей информации с преодолением мер защиты;
- незаконное подключение к аппаратуре и линиям связи;
- злоумышленный вывод из строя механизмов защиты;

- внедрение и использование компьютерных вирусов.

Вопрос 2. Меры противодействия угрозам

К основным средствам защиты относятся:

1. Правовые и законодательные: законы, указы, нормативные акты, регламентирующие правила обращения с информацией и определяющие ответственность за нарушение этих правил.

2. Морально-этические – нормы поведения, которые традиционно сложились или складываются в обществе по мере распространения вычислительной техники. Невыполнение этих норм ведет к падению авторитета, престижа организации, страны, людей.

3. Административные или организационные. Меры организационного характера, регламентирующие процессы функционирования автоматизированных систем, деятельность персонала с целью максимального затруднения или исключения реализации угроз безопасности:

- организация явного или скрытого контроля за работой пользователей;
- организация учета, хранения, использования, уничтожения документов и носителей информации;
- организация охраны и надежного пропускного режима;
- мероприятия, осуществляемые при подборе и подготовке персонала;
- мероприятия по проектированию, разработке правил доступа к информации;
- мероприятия при разработке, модификации технических средств.

4. Физические. Применение разного рода технических средств охраны и сооружений, предназначенных для создания физических препятствий на путях проникновения в систему (замки на дверях, где размещена аппаратура, решетки на окнах, электронно-механическое оборудование охранной сигнализации и др.).

5. Программные средства, специально предназначенные для выполнения функций защиты информации

6. Технические (аппаратные). Основаны на использовании технических устройств, входящих в состав автоматизированных систем и выполняющих функции защиты:

- средства аутентификации;
- аппаратное шифрование;
- другие.

Виды защиты в экономических информационных системах классифицируются по направлениям защиты. К основным из них относятся:

- защита информации от несанкционированного доступа;

- защита информации в системах связи;
- защита юридической значимости электронных документов;
- защита конфиденциальной информации от утечки по каналам побочных электромагнитных излучений и наводок;
- защита информации от компьютерных вирусов и других опасных воздействий по каналам распространения программ;
- защита от несанкционированного копирования и распространения программ и ценной компьютерной информации.

К основным методам защиты информации в экономических информационных систем относятся:

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

2. **Управление доступом** – метод защиты информации путем регулирования использования всех ресурсов информационной системы (элементов баз данных, программных и технических средств). Управление доступом включает следующие функции защиты:

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

3. **Маскировка** – метод защиты информации путем ее криптографического закрытия.

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

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

6. **Побуждение** – такой метод защиты, который побуждает пользователя и персонал системы не нарушать установленные порядки за счет соблюдения сложившихся моральных и этических норм.

Защита от несанкционированного копирования и распространения про-

грамм и ценной компьютерной информации осуществляется с помощью специальных программных средств, подвергающих защищаемые программы и базы данных предварительной обработке (вставка парольной защиты, проверки по обращению к устройствам хранения ключа и ключевым дискетам, блокировка отладочных прерываний, проверка рабочей ПЭВМ по ее уникальным характеристикам и т.д.), которая приводит исполняемый код защищаемой программы и базы данных в состояние, препятствующее его выполнению на «чужих» машинах.

Контроль целостности программного обеспечения проводится с помощью внешних средств (программ контроля целостности) и с помощью внутренних средств (встроенных в саму программу). Внешние средства осуществляют контроль при старте системы, а также при каждом запуске программы на выполнение. Внутренние средства контролируют выполнение программ при каждом запуске на выполнение и состоят в сравнении контрольных сумм отдельных блоков программ с их эталонными суммами.

Противодействие несанкционированному изменению прикладных и специальных программ можно обеспечить разными способами, в частности методом контроля целостности базового программного обеспечения специальными программами.

При защите коммерческой информации пользуются всей совокупностью существующих средств и систем защиты данных. Однако при их выборе следует исходить из сравнительной оценки важности защищаемой информации и ущерба, который может нанести ее утрата.

Для реализации мер безопасности используются различные механизмы криптографии, т.е. науки об обеспечении секретности и подлинности передаваемых сообщений.

Сущность криптографических методов заключается в том, что для предотвращения несанкционированного доступа к какому-либо сообщению оно зашифровывается. Когда санкционированный пользователь получает это сообщение, он дешифрует или раскрывает его посредством обратного преобразования криптограммы.

Криптографическая система основывается на использовании специального алгоритма, который запускается уникальным числом, называемым шифрующим ключом. Для обмена зашифрованными сообщениями, как отправителю, так и получателю необходимо знать правильную ключевую установку и хранить ее в тайне.

Шифрование может быть симметричным и асимметричным: первое основывается на использовании одного и того же секретного ключа для шифрования и дешифрования, второе характеризуется тем, что для шифрования исполь-

зуется один ключ, являющийся общедоступным, а для дешифрования – другой, являющийся секретным.

Наряду с шифрованием используются и другие механизмы безопасности:

- цифровая (электронная) подпись;
- контроль доступа;
- обеспечение целостности данных;
- обеспечение аутентификации;
- управление маршрутизацией;
- арбитраж или освидетельствование.

Механизмы цифровой подписи основываются на алгоритмах ассиметричного шифрования и включают две процедуры: формирование подписи отправителем и ее опознавание (верификацию) получателем.

Механизмы контроля доступа осуществляют проверку полномочий объектов автоматизированных ИТ (программ и пользователей) на доступ к ресурсам сети.

Механизмы обеспечения целостности данных реализуются выполнением взаимосвязанных процедур шифрования и дешифрования отправителем и получателем. Отправитель дополняет передаваемый блок криптографической суммой, а получатель сравнивает ее с криптографическим значением, соответствующим принятому блоку. Несовпадение свидетельствует об искажении информации в блоке.

Механизмы управления маршрутизацией обеспечивают выбор маршрутов движения информации по коммуникационной сети таким образом, чтобы исключить передачу секретных сведений по физически ненадежным каналам.

Механизмы арбитража обеспечивают подтверждение характеристик данных, передаваемых между объектами автоматизированных ИТ, третьей стороной (арбитром).

Вопрос 3. Вредительские программы

Нарушение программной структуры компьютерной системы осуществляется злоумышленниками путем изменения кода штатной программы или внедрения вредительских программ. К **вредительским программам** относятся все программы, которые способны выполнять недекларированные действия или препятствовать выполнению штатных действий, не зависимо от последствий наличия таких программ в компьютерной системе.

Компьютерным вирусом называют небольшую программу, которая способна создавать свои копии, внедрять их в файлы, системные области носителей информации и мешать корректной работе компьютера, разрушать инфор-

мацию на внешних носителях.

Пути проникновения вируса в компьютер – сменные носители информации и компьютерные сети.

Внешние признаки, по которым можно догадаться о присутствии вируса в компьютере:

- неправильная работа ранее успешно работавшей программы;
- невозможность загрузки операционной системы;
- искажение содержимого или полное исчезновение файлов и каталогов;
- изменение даты, времени, размера файлов;
- значительное увеличение количества файлов на дисках;
- уменьшение свободной оперативной памяти;
- вывод незапланированных сообщений, изображений на экран, звуковых сигналов;
- частые «зависания» и сбои в работе компьютера.

Классификация вирусов по некоторым признакам:

1. По способу заражения среды обитания:

- резидентные вирусы оставляют в оперативной памяти свою часть, которая будет активна до выключения или перезагрузки компьютера;

- нерезидентные вирусы не заражают оперативную память и активны ограниченное время.

2. По среде обитания:

- файловые вирусы внедряются в исполняемые программы, когда программа запускается, получают управление, выполняют поиск подходя- щего файла и заражают его, т.е. производят некоторое действие, ради которого были написаны, а затем возвращают управление программе;

- загрузочные вирусы внедряются в программу начальной загрузки операционной системы, которая хранится в загрузочном секторе диска, попа- дают в оперативную память и при заражении выполняют следующие действия:

- делают информацию некоторой области диска недоступной;
- внедряют в загрузочный сектор свою копию и пересылают модифицированный код в выделенную область диска;
- организуют передачу управления сначала вирусу и только потом программе начальной загрузки;
- кроме того, вирус может исказить таблицу разделов жесткого диска;

- файлово-загрузочные имеют признаки файловых и загрузочных вирусов;

- драйверные вирусы заражают системную программу – драйвер устройства или включают в файл конфигурации дополнительные строки;

- макровирусы обитают в офисных документах (текстовых и табличных), в

создании которых использовался язык макрокоманд;

- сетевые вирусы используются для своего распространения протоколы компьютерных сетей, и попав на сервер или рабочую станцию, провоци- руют пользователя на запуск объекта, где он содержится.

3. По виду алгоритма, используемого при написании вирусов:

- паразитические вирусы – изменяют содержимое файлов и системных секторов, могут быть легко обнаружены и уничтожены;

- вирусы-репликаторы (черви) распространяются по компьютерным се- тям, вычисляют адреса сетевых компьютеров и рассылают по этим адре- сам свои копии;

- вирусы-невидимки (стелс-вирусы) трудно обнаружить и обезвредить, так как они перехватывают обращение к секторам и подменяют их неза- раженными участками;

- вирусы-мутанты содержат алгоритмы шифровки-расшифровки, копии вируса не содержат повторяющихся частей и их трудно обнаружить;

- троянские кони не способны к самораспространению, маскируются под полезную программу и разрушают загрузочный сектор или файловую систему.

Вопрос 4. Программные средства защиты информации

Наибольшую опасность целостности информации ПК представляют пред- намеренные угрозы, создаваемые людьми. Противостоять этим угрозам можно, используя следующие методы.

1. Аутентификация пользователя предлагает применение пароля, опозна- вание в диалоговом режиме, применение физиологических характеристик лич- ности, радиокодовых устройств, электронных карточек.

Каждому зарегистрированному пользователю выдается пароль. Программа сравнивает введенный пароль с эталоном и при совпадении предоставляет пользователю доступ к системе.

Для опознавания в диалоговом режиме в файлах защиты создаются записи, содержащие данные о личности пользователя (дата рождения, рост, имена и да- ты рождения родственников и т.д.) или большой набор паролей. При обраще- нии пользователя программа механизма защиты предлагает пользователю на- звать некоторые данные, которые сравниваются с имеющимися в системе запи- сями. По результатам сравнения принимается решение о доступе. Для повыше- ния надежности при каждом входе в систему запрашиваются другие данные.

Для опознания по индивидуальным физиологическим характеристикам,

например, по отпечаткам пальцев, требует специальную аппаратуру и сложенные программы для обработки этих параметров и сравнения с эталоном. Поэтому этот способ редко применяется в компьютерных системах. Возможно опознавание пользователя по параметрам его работы с клавиатурой – скорость набора текста, интервалы между нажатием клавиш носят сугубо индивидуальный характер.

Опознавание по радиокодовым устройствам предоставляет доступ владельцу устройства, генерирующего индивидуальный радиосигнал. ПК оснащается программно-аппаратными средствами для приема, регистрации и обработки сигнала. Но если устройство будет украдено, то его новый владелец может войти в систему.

Опознавание по карточкам более надежно. На карточку заносят персональную информацию о пользователе в зашифрованном виде, информация и шифр известны только пользователю и системе.

2. Разграничение доступа к информации означает, что зарегистрированному пользователю предоставляется доступ в пределах его полномочий. Разработаны методы разграничения доступа к устройствам компьютера, к программам, к областям запоминающих устройств, к базам данных. Разграничение может проводиться несколькими способами:

- по кольцам секретности – данные распределяются по массивам так, чтобы в каждом массиве содержались данные одного уровня секретности, например, «секретно» или «совершенно секретно», а пользователю предоставляется только один уровень допуска и запрещен доступ к массиву другого уровня;

- по спискам – для файла составляется список пользователей, которые имеют к нему доступ;

- по матрицам полномочий – формулируется матрица, строки которой содержат имена пользователей, а столбцы – элементы данных, в которых содержится информация о доступе пользователя к соответствующей информации;

- по мандатам – элементам информации присваиваются метки, и пользователю предоставляется разовое разрешение на доступ, если он предъя- вит метку элемента.

3. Криптографическая защита информации, предназначенной для длительного хранения. Шифрование данных совмещается с архивацией, сжатием. Иногда криптографическая защита применяется к информации в процессе ее обработки.

4. Регистрация обращений к защищаемой информации позволяет повысить эффективность защиты, присутствует во всех системах защиты информации.

Для борьбы с компьютерными вирусами наиболее часто применяются ан-

тивирусные программы. Эти программы можно разделить на несколько видов:

- **программы-детекторы** позволяют обнаруживать файлы, зараженные одним из нескольких известных вирусов;

- **программы-доктора**, или **фаги**, «лечат» зараженные программы или диски, «выкусывая» из зараженных программ тело вируса, т.е. восстанавливая программу в том состоянии, в котором она находилась до заражения вирусом. В начале своей работы фаги ищут вирусы в оператив- ной памяти, уничтожая их, и только затем переходят к «лечению» фай- лов. Среди фагов выделяют полифаги, т.е. программы-доктора, предна- значенные для поиска и уничтожения большого количества вирусов. Наиболее известными полифагами являются программы *Aids test*, *Scan*, *Norton Antivirus* и *Doctor Web*;

- **программы-ревизоры** сначала запоминают сведения о состоянии программ и системных областей дисков, а затем сравнивают их состояние с исходным. При выявлении несоответствий об этом сообщается пользо- вателю. Программы-ревизоры имеют достаточно развитые алгоритмы, обнаруживают стелс-вирусы и могут даже отличить изменения версии проверяемой программы от изменений, внесенных вирусом. К числу программ-ревизоров относится широко распространенная в России про- грамма *A Dinf* «Диалог-Наука»;

- **доктора-ревизоры** – это гибриды ревизоров и докторов, т.е. программы, которые не только обнаруживают изменения в файлах и системных об- ластях дисков, но и могут в случае изменений автоматически вернуть их в исходное состояние;

- **программы-фильтры** располагаются резидентно в оперативной памяти компьютера и перехватывают те обращения к операционной системе, которые используются вирусами для размножения и нанесения вреда, и сообщают о них пользователю. Пользователь может разрешить или за- претить выполнение соответствующей операции;

- **программы-вакцины**, или иммунизаторы, модифицируют программы и диски таким образом, что это не отражается на работе программ, но тот вирус, от которого производится вакцинация, считает эти программы или диски уже зараженными. Эти программы крайне неэффективны.

Рассмотрим некоторые современные антивирусные программы.

«Доктор Веб» – российский производитель антивирусных средств защиты информации под маркой *Dr.Web*. ***Dr. Web*** (рус. Доктор Веб) – семейство антивирусов, предназначенных для защиты от почтовых и сетевых червей, руткитов, файловых вирусов, троянских программ, стелс-вирусов, полиморфных вирусов, бестелесных вирусов, макровирусов, вирусов, поражающих документы MS Office, скрипт-вирусов, шпионского ПО (spyware), программ-похитителей

паролей, клавиатурных шпионов, программ платного дозвона, рекламного ПО (adware), потенциально опасного ПО, хакерских утилит, программ-люков, программ-шуток, вредоносных скриптов и других вредоносных объектов, а также от спама, скаминг-, фарминг-, фишинг-сообщений и технического спама.

«Доктор Веб» стал первой компанией, предложившей на российском рынке инновационную модель использования антивируса в качестве услуги, и по сей день продолжает оставаться безусловным лидером российского рынка интернет-сервисов безопасности для поставщиков ИТ-услуг.

Выгоды использования продуктов компании «Доктор Веб»:

- полная защита от существующих угроз;
- увеличение производительности труда сотрудников за счет снижения потока спама и исключения простоев ПК сотрудников во время восстановления информации;
- сохранение репутации компании, сети которой не будут использоваться как источник вирусов и спама, которые могут попастьклиентам;
- снижение непрофильных затрат за счет снижения нагрузки на системных администраторов, уменьшения вынужденных простоев сотрудников и предприятия в целом, уменьшение стоимости закупаемых серверов и рабочих станций;
- простота выбора нужной конфигурации, так как для продуктов компании «Доктор Веб» используется гибкая система лицензирования.

Среди потребителей продуктов компании – домашние пользователи из всех регионов мира и крупные российские предприятия, небольшие организации и системообразующие корпорации. На основе продуктов компании «Доктор Веб» построены системы информационной безопасности практически всех крупных органов государственной власти России – Правительства РФ, Совета Федерации РФ, Администрации Президента РФ, Министерства обороны РФ, Министерства иностранныхдел РФ, ФСБ, Министерства финансов РФ, Министерства образования и науки РФ и многих других госучреждений. Система *Dr.Web Enterprise Security Suite* защищает сети многих крупных организаций с территориальнораспределенной структурой.

AVG Antivirus – антивирусная система производства Чехии, имеющая сканер файлов, сканер электронной почты и поддерживающая возможность автоматического наблюдения. *AVG Antivirus* предлагает защиту от самых сложных на сегодняшний день угроз. Безопасная загрузка и обмен файлами, безопасный чат, игры и просмотр фильмов без прерывания. Загрузка, обмен файлами и отправка сообщений безопасно с *AVG Online Shield*.Безопасное пребывание в социальных сетях с *AVG Social Networking Protection*. Заходите на веб-сайты и ищите информацию в Интернете с уверенностью под защитой *AVG LinkScanner*

в режиме реального времени.

Антивирус *AVG* существенно не влияет на производительность системы – надежное, быстрое и легкое средство защиты компьютера от вирусов и других вредоносных программ. Он выполняет обнаружение вирусов, интернет-червей, троянских программ, а также нежелательных файлов или библиотек в системе. Для поддержания максимально возможного уровня защиты компонент *AVG Antivirus* необходимо постоянно обновлять.

AVG защищает компьютер от вредоносного и рекламного ПО, обеспечивая защиту от нежелательных рекламных объявлений и ПО, тайно собирающего личные данные с компьютера.

Компонент *AVG* «Хранилище данных» позволяет создавать надежные виртуальные хранилища для ценных и конфиденциальных данных. Содержимое **Хранилище данных** шифруется и защищается заданным паролем, таким образом, предотвращая неавторизованный доступ.

AVGE-mail Scanner проверяет входящие и исходящие сообщений электронной почты с помощью подключаемых модулей, разработанных для наиболее распространенных клиентов электронной почты.

Антивирус Касперского (*Kaspersky Antivirus*, *KAV*) – антивирусное программное обеспечение, разрабатываемое Лабораторией Касперского. Предоставляет пользователю защиту от вирусов, троянских программ, шпионских программ, руткитов, adware, а также неизвестных угроз с помощью проактивной защиты, включающей компонент *HIPS*.

Лаборатория Касперского – самый известный в России производитель систем защиты от вредоносных программ, спама и хакерских атак и крупнейшая антивирусная компания в Европе, входит в четверку ведущих мировых производителей программных решений для обеспечения информационной безопасности для конечных пользователей.

Антивирус Касперского – это программное обеспечение, предназначенное для базовой защиты компьютера от разного рода вредоносных программ. Приложение обеспечивает комплексную защиту компьютера в режиме реального времени от как известных, так и новых информационных угроз.

Основными преимуществами данной антивирусной программы являются:

- мгновенная реакция на нововыявленные угрозы и попытки нанесения вреда компьютеру, так как угрозы блокируются при помощи информации из *Kaspersky Security Network*;

- гибридный способ защиты от вредоносного программного обеспечения дает возможность достижения максимального уровня безопасности, оптимально используя его ресурсы при помощью комбинации антивирусных и сетевых технологий;

- проверка уровня репутации интернет-сайтов и приложений – антивирусом определяется степень опасности подозрительных приложений еще до запуска их на компьютере и выявляет опасные сайты еще до того как пользователь перейдет на них;

- конструктивно понятно реализованный внешний вид дает возможность получения быстрого доступа ко всем возможностям и функциям приложения;

- благодаря применению новинок области сетевых технологий, мониторингом активности приложений отслеживается и анализируется больше событий на компьютере;

- анти-фишингом пользователь эффективно защищается от интернет-мошенничества, обращаясь к *Kaspersky Security Network* в режиме ре- альном времени;

- проверка репутации приложений дает возможность получения сведений о исполняемых файлах и принимать решения, стоит ли использовать его;

- интеллектуальная система обновлений способна загружать обновления только активных компонентов программы, что значительно экономит сетевой трафик и оперативную память компьютера.

Список использованной литературы

1. Microsoft Word. Версия 2002. Шаг за шагом: практическое пособие: [пер. с англ.]. – М.: ЭКОМ, 2012. – 336 с.
2. Безручко, В.Т. Практикум по курсу «Информатика». Работа в Windows, Word, Excel: учеб. пособие / В.Т. Безручко. – 2-е изд. – ФГУП. Издательский дом «Финансы и статистика», 2015. – 544 с.
3. Гаврилов, М.В. Информатика и информационные технологии: учебник/ М.В. Гаврилов. – М.: Гардарики, 2017. – 655 с.
4. Гуда, А.Н. Информатика: учебник / А.Н. Гуда [и др.]; под ред. В.И. Колесникова. – М.: Ростов н/Д: Дашков и К, 2019. – 400 с.
5. Гукин, Д. MicrosoftWord для Windows “чайников”: [пер. с англ.] / Д.Гукин. – М.: Издательский дом «Вильямс», 2013. – 336 с.
6. Горяев, Ю.А. Информатика: учеб. пособие / Ю.А. Горяев. – М.: МИЭМП, 2015. – 116 с.
7. Додж, М. Эффективная работа с Microsoft Excel: [пер. с англ.] | М.Додж, К. Стинсон. – СПб.: Питер, 2013. – 1056 с.
8. Догадин, Н.Б. Архитектура компьютера: учеб. пособие / Н.Б. Догадин. – М.: Бином. «Лаборатория знаний», 2018. – 271 с.
9. Зегжда, Д.П. Основы безопасности информационных систем / Д.П. Зегжда, А.М. Ивашко. – М.: Горячая линия - Телеком, 2017. – 452 с.
10. Информатика. Базовый курс. Учебник. Под ред. Симоновича С. В. Рекомендовано Министерством образования и науки РФ. СПб.: Питер, 2018. – 640 с.
11. Калабухова, Г.В. Компьютерный практикум по информатике. Офисные технологии: учеб. пособие / Г.В. Калабухова, В.М. Титов. – М.: ФОРУМ, 2018. – 336 с.
12. Калугина, О.Б. Работа с текстовой информацией / О.Б. Калугина, В.С. Люцарев. – М.: Интернет-Университет Информационных Технологий, 2015. – 152 с.
13. Кокс, Д. Microsoft Office Power Point. Шаг за шагом: [пер. с англ.] / Д. Кокс, Д. Преппернау. – М.: ЭКОМ Паблишерз, 2017. – 448 с
14. Леонтьев, В.П. Интернет / В.П. Леонтьев. – М.: ОЛМА-ПРЕСС Образование, 2016. – 47 с.
15. Майстренко, А.В. Информатика: учеб. пособие / А.В. Майстренко. – Тамбов: Изд-во ТГТУ, 2016. – 80 с.
16. Макарова, Н.В. Информатика: учебник для вузов / Н.В. Макарова, В.Б. Волков. – СПб: Питер, 2018. – 576 с.
17. Марков, А.С. Базы данных. Введение в теорию и методологию: учебникт / А.С. Марков, К.Ю. Лисовский. – М.: Финансы и статистика, 2018. – 512 с.

18. Мельников, В.П. Информационная безопасность и защита информации: учеб. пособие для вузов / В. П. Мельников, С. А. Клейменов; под ред. С.А. Клейменова. – М.: Академия, 2016. – 336 с.
19. Моисеенко, Е.В. Информационные технологии в экономике: учеб. пособие / Е.В. Моисеенко, Е.Г. Лаврушина. – Владивосток: Изд-во ВГУЭС, 2017. – 246 с.
22. Назаров, С.В. Программирование на MS Visual Basic: учеб. пособие / С.В.Назаров, П.П. Мельников. – М.: Финансы и статистика, 2018. – 320 с.
23. Олифер, В.Г. Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов / В.Г.Олифер, Н.А. Олифер. – СПб: Питер, 2019. – 944 с.
24. Партыка, Т.Л. Операционные системы, среды и оболочки: учеб. пособие. / Т.Л.Партыка, И.И. Попов. – М.: ФОРУМ, 2018. – 544 с.
25. Перри, Г. Самоучитель программирования / Г. Перри. –1-е изд. – СПб: Питер, 2019. – 368 с.
26. Пикуза, В. Экономические и финансовые расчеты в Excel. (+ дискета). Самоучитель / В. Пикуза, А. Гаращенко. – СПб.: Питер, 2017. – 397 с.
27. Райкин, И.Л. Информационная безопасность и защита информации: учебник для вузов / И.Л. Райкин; НГТУ. – Н. Новгород, 2019. – 256 с.
28. Савицкий, Н. И. Экономическая информатика: учеб. пособие для вузов / Н. И. Савицкий. – М.: Экономистъ, 2016. – 429 с.
29. Сайлер, Б. Использование Visual Basic. Специальное издание:[пер. с англ.] / Б. Сайлер, Д. Споттс. – М.: Издательский дом «Вильямс», 2018. – 832 с.
30. Таненбаум, Э. Компьютерные сети / Э. Таненбаум. – СПб.: Питер, 2018. – 992 с.
31. Таненбаум, Э. Архитектура компьютера / Э. Таненбаум. – СПб.: Питер, 2017. – 704 с.
32. Титоренко, Г.А. Информационные системы в экономике: учеб. пособие / Г.А. Титоренко. – 2-е изд. перераб. и дополн. – М.: Юнити-Дана, 2018. – 463 с.
33. Халилов, М.С. Информатика: учебник / М.С. Халилов. Издательство БГУ, 2018. – 368 с.