ИНФОРМАТИКА
Часть 2

Зобнин Юрий Александрович
кандидат социологических наук
доцент кафедры бизнес-информатики и математики
каб. 519 (7-й корп. ТИУ)
greentree1959@mail.ru

Программное обеспечение

- Базовое, Системное, Служебное и Прикладное ПО
- Операционная система Windows

Решение задач на ЭВМ

- Этапы решения задачи на ЭВМ
- Моделирование и модели
- Классификация моделей
- Построение модели
- Классификация моделирования

Величины и алгоритмы

- Величины и их свойства
- Алгоритмы: понятие, свойства, формы представления
- Элементы блок-схемы алгоритма
- Основные виды алгоритмов

Основы программирования

- Языки программирования
- Технологии программирования

А.С. Пушкин: «Детина полоумный на диване…»

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Программы — это упорядоченная последовательность команд. Конечная цель любой программы - управление аппаратными средствами. Даже если на первый взгляд программа никак не взаимодействует с оборудованием, не требует никакого ввода данных с устройств ввода и не осуществляет вывод данных на устройства вывода, все равно ее работа основана на управлении аппаратными устройствами компьютера.

Программное и аппаратное обеспечение в компьютере работают в неразрывной связи и в непрерывном взаимодействии.

- Прикладное ПО
- Служебное ПО
- Системное ПО
- Базовое ПО

Уровни (виды) ПО представляют собой пирамидоидальную конструкцию. Каждый следующий уровень опирается на ПО предшествующих уровней. Такое разделение удобно для всех этапов работы с вычислительной системой, начиная с установки программ до практической эксплуатации и технического обслуживания. Каждый вышележащий уровень повышает функциональность всей системы. Так, например, вычислительная система с ПО базового уровня не способна выполнять большинство функций, но позволяет установить системное ПО.

БАЗОВЫЙ УРОВЕНЬ - самый нижний уровень – представляет базовое ПО (BIOS – basic input-output sistem). Оно отвечает за взаимодействие с базовыми аппаратными средствами. Как правило, базовые программные средства непосредственно входят в состав базового оборудования и хранятся в специальных микросхемах, называемых постоянными запоминающими устройствами (ПЗУ - Read Only Memory, ROM). Программы и данные записываются (прошиваются) в микросхемах ПЗУ на этапе производства и не могут быть изменены в процессе эксплуатации.

СИСТЕМНЫЙ УРОВЕНЬ - переходный. Программы, работающие на этом уровне, обеспечивают взаимодействие прочих программ компьютерной системы с программами базового уровня и непосредственно с аппаратным обеспечением, то есть выполняют «посреднические» функции (пример: операционные системы, драйверы).

От ПО системного уровня зависят эксплуатационные показатели всей вычислительной системы в целом. Так, например, при подключении к вычислительной системе нового оборудования на системном уровне должна быть установлена программа, обеспечивающая для других программ взаимосвязь с этим оборудованием. Конкретные программы, отвечающие за взаимодействие с конкретными устройствами, называются драйверами устройств - они входят в состав ПО системного уровня. Не требуют драйверов дисководы, клавиатура.

Другой класс программ системного уровня отвечает за взаимодействие с пользователем. Именно благодаря им он получает возможность вводить данные в вычислительную

Совокупность ПО системного уровня образует «ядро операционной системы компьютера». Если компьютер оснащен ПО системного уровня, то он уже подготовлен к установке программ более высокого уровня, к взаимодействию программных средств с оборудованием и к взаимодействию с пользователем. То есть наличие ядра операционной системы - непременное условие для возможности практической работы человека с вычислительной системой.

СЛУЖЕБНЫЙ УРОВЕНЬ - ПО этого уровня взаимодействует как с программами базового уровня, так и с программами системного уровня. Основное назначение служебных программ (их также называют утилитами) состоит в автоматизации работ по проверке, наладке и настройки компьютерной системы. Во многих случаях они используются для расширения или улучшения функций системных программ, некоторые служебные программы (как правило, это программы обслуживания) изначально включают в состав операционной системы, но большинство служебных программ являются для операционной системы внешними и служат для расширения ее функций. К таким программам относятся служебные программы операционной системы, архиваторы, антивирусные программы.

Служебные программные средства

Диспетчеры файлов (файловые менеджеры). С помощью программ этого класса выполняется большинство операций, связанных с обслуживанием файловой структуры: копирование, перемещение и переименование файлов, создание каталогов (папок), удаление файлов и каталогов, поиск файлов и навигация в файловой структуре. Базовые программные средства, предназначенные для этой цели, обычно входят в состав программ системного уровня и устанавливаются вместе с операционной системой. Однако для повышения удобства работы с компьютеров, большинство пользователей устанавливают дополнительные служебные программы.

Средства сжатия данных (архиваторы). Создание архивов упрощает хранение данных за счет того, что большие группы файлов и каталогов сводятся в один архивный файл. При этом повышается и эффективность использования носителя за счет того, что архивные файлы обычно имеют более повышенную плотность записи информации. Архиваторы часто используют для создания резервных копий ценных данных.

Средства просмотра и воспроизведения. Обычно для работы с файлами данных необходимо загрузить их в «родительскую» прикладную систему, в которой они были созданы. Это дает возможность просматривать документы и вносить в них изменения. Но в тех случаях, когда требуется просматривать документы, без их редактирования, удобно использовать более простые и более универсальные средства, позволяющие просматривать документы разных типов.

Средства диагностики предназначены для автоматизации процессов диагностики программного и аппаратного обеспечения. Они выполняют необходимые проверки и выдают собранную информацию в удобном и наглядном виде. Их используют не только для устранения неполадок, но и для оптимизации работы компьютерной системы.

Средства контроля (мониторинга) позволяют следить за процессами, происходящими в компьютерной системе. При этом возможны два подхода: наблюдение в реальном времени или контроль с записью результатов в специальном протокольном файле. Первый подход используют при изыскании путей для оптимизации работы вычислительной системы и повышении ее эффективности. Второй подход используется, когда мониторинг выполняется автоматически и (или) дистанционно. В последнем случае результаты мониторинга можно передать удаленной службе технической поддержки для установления причин конфликта в работе программного и аппаратного обеспечения.

Мониторы установки предназначены для контроля за установкой ПО. Необходимость в этих средствах связана с тем, что между различными категориями ПО могут устанавливаться вертикальные и горизонтальные связи.

Мониторы установки следят за изменением и состоянием окружающей программной среды, отслеживают и протоколируют образование новых связей, и позволяют восстанавливать связи, утраченные в результате удаления ранее установленных программ. Простейшие средства управления установкой и удалением программ обычно входят в состав операционной системы и размещаются на системном уровне программного обеспечения.

Средства коммуникации (коммуникационные программы) позволяют устанавливать соединения с удаленными компьютерами, обслуживают передачу сообщений электронной почты, обеспечивают пересылку факсимильных сообщений и выполняют множество других операций в компьютерных сетях.

Средства обеспечения компьютерной безопасности - средства активной и пассивной защиты данных от повреждения, а также средства защиты от несанкционированного доступа, просмотра и изменения данных. В качестве средств пассивной защиты используют служебные программы, предназначенные для резервного копирования. В качестве средств активной защиты применяют антивирусное ПО. Для защиты данных от несанкционированного доступа, их просмотра и изменения служат специальные системы, основанные на криптографии.

ПРИКЛАДНОЙ УРОВЕНЬ - 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. Огромный функциональный диапазон возможных приложений средств вычислительной техники обусловлен наличием прикладных программ для разных видов деятельности.

Поскольку между прикладным ПО и системным существует непосредственная взаимосвязь (первое опирается на второе), то можно утверждать, что универсальность вычислительной системы, доступность прикладного ПО и широта функциональных возможностей компьютера напрямую зависят от типа используемой операционной системы, от того, какие системные средства содержит ее ядро, как она обеспечивает взаимодействие триединого комплекса: человек - программа - оборудование.

Прикладные программные средства

Текстовые редакторы. Основные функции этого класса прикладных программ - ввод и редактирование текстовых данных. Дополнительные функции - автоматизация процессов ввода и редактирования. Для операций ввода, вывода и сохранения данных текстовые редакторы вызывают и используют системное ПО.

С этого класса прикладных программ обычно начинают знакомство с ПО и на нем отрабатывают первичные навыки взаимодействия с компьютерной системой.

Текстовые процессоры. Основное отличие текстовых процессоров от редакторов в том, что они позволяют не только вводить и редактировать текст, но и форматировать его, то есть оформлять. Соответственно, к основным средствам текстовых процессоров относятся средства обеспечения взаимодействия текста, графики, таблиц и других объектов, составляющих итоговый документ, а к дополнительным - средства автоматизации процесса форматирования.

Современный стиль работы с документами подразумевает два альтернативных подхода – работу с бумажными документами и работу с электронными документами (по безбумажной технологии). Поэтому, говоря о форматировании документов средствами текстовых процессоров, надо иметь в виду два принципиально разных направления - форматирование документов, предназначенных для печати, и форматирование документов, предназначенных для отображения на экране. Приемы и методы в этих случаях существенно различаются. Соответственно, различаются и текстовые процессоры, хотя многие из них успешно сочетают оба подхода.

Графические редакторы - обширный класс программ, предназначенных для создания и (или) обработки графических изображений. В данном классе различают следующие категории: растровые редакторы, векторные редакторы и программные средства для создания и обработки трехмерной графики (3Д-редакторы).

Растровые редакторы применяются в тех случаях, когда графический объект представлен в виде комбинации точек, образующих растр и обладающих свойствами яркости и цвета.) Такой подход эффективен в тех случаях, когда графическое изображение имеет много полутонов и информация о цвете элементов, составляющих объект, важнее, чем информация об их форме. Это характерно для фотографических и полиграфических изображений.

Растровые редакторы широко применяются для обработки изображений, их ретуши, создания

Возможности создания новых изображений средствами растровых редакторов ограничены и не всегда удобны. В большинстве случаев художники предпочитают пользоваться традиционными инструментами, после чего вводить рисунок в компьютер с помощью специальных аппаратных средств (сканеров) и завершать работу с помощью растрового редактора путем применения спецэффектов.

Векторные редакторы отличаются от растровых способом представления данных об изображении. Элементарным объектом векторного изображения является не точка, а линия. Такой подход характерен для чертежно-графических работ, в которых форма линий имеет большее значение, чем информация о цвете отдельных точек, составляющих ее. В векторных редакторах каждая линия рассматривается как математическая кривая третьего порядка и, соответственно, представляется не комбинацией точек, а формулой (в компьютере хранятся числовые коэффициенты этой формулы).

Векторное представление намного компактнее, чем растровое, поэтому данные занимают много меньше места, однако построение любого объекта заполняется не простым отображением точек на экране, а сопровождается непрерывным пересчетом параметров кривой в координаты экранного или печатного изображения. Соответственно, работа с векторной графикой требует более производительных вычислительных систем.

Из элементарных объектов (линий) создаются простейшие геометрические фигуры (примитивы) из которых, в свою очередь, составляются законченные композиции. Художественная иллюстрация, выполненная средствами векторной графики, может содержать десятки тысяч простейших объектов, взаимодействующих друг с другом.

Векторные редакторы удобны для создания изображений, но практически не используются для создания готовых рисунков. Они нашли широкое применение в рекламном бизнесе, их применяют для оформления обложек полиграфических изданий и всюду, где стиль художественной работы близок к чертежному.

Редакторы трехмерной графики используются для создания трехмерных композиций. Они имеют две характерные особенности. Во-первых, они позволяют гибко управлять взаимодействием свойств поверхностей со свойствами источников освещения и, во-вторых, позволяют создавать трехмерную анимацию. Поэтому редакторы трехмерной графики нередко называют также 3D-аниматорами.

Системы управления базами данных (СУБД). Базами данных называют огромные массивы данных, организованных в табличные структуры. Основными функциями систем управления базами данных являются:

- создание пустой (незаполненной) структуры базы данных;
- предоставление средств ее заполнения или импорта данных из таблиц другой базы данных;
- обеспечение возможности доступа к данным, а также предоставление средств поиска и фильтрации.

Многие системы управления базами данных дополнительно предоставляют возможности проведения простейшего анализа данных и их обработки. В результате возможно создание новых таблиц баз данных на основе имеющихся. В связи с широким распространением сетевых технологий к современным системам управления базами данных предъявляется также требование возможности работы с удаленными и распределенными ресурсами, находящихся на серверах всемирной компьютерной сети.

Электронные таблицы предоставляют комплексные средства для хранения различных типов данных и их обработки. В некоторой степени они аналогичны системам управления базами данных, но основной акцент смещен не на хранении массивов данных и обеспечении к ним доступа, а на преобразовании данных, причем в соответствии с их внутренним содержанием.

В отличии от баз данных, которые обычно содержат широкий спектр типов данных (от числовых и текстовых до мультимедийных), для электронных таблиц характерна повышенная сосредоточенность на числовых данных. Зато электронные таблицы предоставляют более широкий спектр методов для работы с числовыми данными.

Основное свойство электронных таблиц состоит в том, что при изменении содержания любых ячеек таблицы может происходить автоматическое изменение во всех прочих ячейках, связанных с измененными соотношением, заданным математическими или логическими выражениями (формулами). Простота и удобство работы с электронными таблицами снискали им широкое применение в сфере бухгалтерского учета, в качестве универсальных инструментов анализа финансовых, сырьевых и товарных рынков, доступных средств обработки результатов технических испытаний, то есть повсюду, где необходимо автоматизировать регулярно повторяющиеся вычисления достаточно больших объемов числовых данных.

Экспертные системы предназначены для анализа данных, содержащихся в базах знаний, и выдачи рекомендаций по запросу пользователя. Такие системы применяют в тех случаях, когда исходные данные хорошо формализуются, но для принятия решения требуются обширные специализированные знания. Характерными областями использования экспертных систем являются медицина, юриспруденция, фармакология, химия. По совокупности признаков заболевания медицинские экспертные системы помогают установить диагноз и назначить лекарства, дозировку и программу лечебного курса. По совокупности признаков события юридические экспертные системы могут дать правовую оценку и предложить порядок действия, как для обвиняющей стороны, так и для защищающейся.

Бухгалтерские системы - специализированные системы, сочетающие в себе функции текстовых и табличных редакторов, электронных таблиц и систем управления базами данных. Предназначены для автоматизации подготовки первичных бухгалтерских документов предприятия и их учета, для ведения счетов бухгалтерского учета, а также для автоматической подготовки регулярных отчетов по итогам производственной, хозяйственной и финансовой деятельности в форме, принятой для предоставления в налоговые органы, внебюджетные фонды и органы статистического учета. Несмотря на то, что теоретически все функции, характерные для бухгалтерских систем, можно исполнять и другими вышеперечисленными программными средствами, использование бухгалтерских

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

Настольные издательские системы – предназначены для автоматизации процесса верстки полиграфических изданий. Этот класс ПО обеспечения занимает промежуточное значение между текстовыми процессорами и системами автоматизированного проектирования.

ОПЕРАЦИОННЫЕ СИСТЕМЫ

Операционная система (ОС) - программа, загружаемая автоматически при включении компьютера. Осуществляет диалог с пользователем, организует управление ресурсами компьютера и выполнение других программ. Представляет собой комплекс **системных и служебных** программных средств. С одной стороны она опирается на базовое ПО компьютера, входящее в его систему BIOS, с другой стороны она сама является опорой для ПО более высоких уровней – прикладных и большинства служебных приложений.

Приложениями операционной системы принято называть программы, предназначенные для работы под управлением данной системы. Основная функция всех операционных систем – посредническая. Она заключается в обеспечении нескольких видов интерфейса:

- интерфейс пользователя (между пользователями и программно-аппаратными средствами компьютера);
- аппаратно-программный интерфейс (между программным и аппаратным обеспечением);
- программный интерфейс (между различными видами программного обеспечения).

Все операционные системы способны обеспечивать как пакетный, так и диалоговый режим работы с пользователем. В пакетном режиме операционная система автоматически исполняет заданную последовательность команд. В диалоговом режиме операционная система находится в ожидании команды пользователя и получив ее приступает к исполнению.

По реализации интерфейса пользователя различают неграфические и графические операционные системы.

Неграфические операционные системы реализуют интерфейс командной строки. Основным устройством управления в данном случае является клавиатура. Управляющие команды вводят в поле командной строки, где их можно и редактировать. Исполнение команды начинается после ее утверждения, например, нажатием клавиши ENTER. Интерфейс командной строки обеспечивается семейством операционных систем под общим названием MS-DOS.

Графические операционные системы (Windows) реализуют интерфейс в котором в качестве органа управления кроме клавиатуры может использоваться мышь или адекватное устройство позиционирования. Работа с графической операционной системой основана на взаимодействии активных и пассивных элементов управления. В качестве активного элемента управления выступает указатель мыши. В качестве пассивных элементов управления выступают графические элементы управления приложений (экранные кнопки, значки, переключатели, флажки, раскрывающиеся списки, строки меню и другие).

Все операционные системы обеспечивают свой автоматический запуск. Для дисковых операционных систем в специальной системной области диска создается запись программного кода. Обращение к этому коду выполняют программы, находящиеся в базовой системе ввода – вывода. Завершая свою работу, они дают команду на загрузку и исполнение содержимого системной области диска.

Все современные дисковые операционные системы обеспечивают создание файловой системы, предназначенной для хранения данных на дисках и обеспечения доступа к ним. Принцип организации файловой системы - табличный. Поверхность жесткого диска рассматривается как трехмерная матрица, измерениями которой являются номера поверхности, цилиндра и сектора. Данные о том, в каком месте диска записан тот или иной файл, хранятся в системной области диска в специальных таблицах размещения файлов (FAT – таблицах). Поскольку нарушение FAT – таблицы приводит к невозможности воспользоваться данными, записанными на диске, к ней предъявляются особые требования надежности, и она существует в двух экземплярах, идентичность которых

Пользователю данные представляются в виде иерархической структуры. К функциям обслуживания файловой структуры относятся следующие операции, происходящие под управлением операционной системы:

- создание файлов и каталогов (папок), присвоение им имен
- переименование файлов и каталогов
- копирование и перемещение файлов между дисками компьютера и между каталогами (папками) одного диска
- удаление файлов и каталогов (папок)
- навигация по файловой структуре с целью доступа к заданному файлу, каталогу (папке)
- управление атрибутами файлов.

Атрибуты – это дополнительные параметры, определяющие свойства файлов. Основных атрибутов четыре:

1. «Только для чтения» ограничивает возможности работы с файлом. Его установка означает, что файл не предназначен для внесения изменений.
2. «Скрытый» означает, что данный файл не следует отображать на экране при проведении файловых операций. Это мера защиты против случайного (умышленного или неумышленного) повреждения файла.
3. «Системный» – помечаются файлы, обладающие важными функциями в работе самой операционной системы. Его отличительная особенность в том, что средствами операционной системы его изменить нельзя.
4. «Архивный» - в прошлом использовался для работы программ резервного копирования. В настоящее время данный атрибут во внимание не принимается.

С точки зрения управления исполнением приложений различаются однозадачные и многозадачные операционные системы. Одной из важных функций операционных систем является предоставление основных средств обслуживания компьютера. Обычно это решается путем включения в базовый состав операционной системы служебных приложений: средств проверки диска, средств сжатия диска, средства управления виртуальной памятью, средства кэширования дисков, средства резервного копирования данных.

ОПЕРАЦИОННАЯ СИСТЕМА WINDOWS

Основными достоинствами Windows являются: многозадачность, объектно-ориентированный подход, графический пользовательский интерфейс, а также простое подключение к компьютеру новых устройств.

Многозадачность обеспечивает возможность запуска и работы сразу с несколькими приложениями.

Объектно-ориентированный подход позволяет рассматривать элементы операционной системы (папки, окна, приложения, документы и т.д.) как объекты. Объект характеризуется свойствами и операциями, которые над ним можно осуществить. Например, документы (документом называется любой файл, обрабатываемый с помощью приложений) имеют определенный объем и их можно копировать, перемещать и переименовывать, окна имеют размер, который можно изменять и т.д. Хотя каждый из этих объектов имеет свои конкретные свойства и над ним возможны определенные операции, технология работы с объектами и интерфейс универсальны. Это позволяет пользователю достичь единообразия при работе с разными объектами.

Объектно-ориентированный подход позволяет реализовать механизм встраивания и внедрения объектов **OLE (Object Linking Embedding)**. Этот механизм позволяет копировать и вставлять объекты из одного приложения в другое. Например, работая с документом в текстовом редакторе Word, в него можно встроить изображения, звук и даже видеофрагменты и таким образом из обычного текстового документа получить мультимедиа документ. Графический пользовательский интерфейс позволяет унифицировать и сделать наглядной работу с объектами. Основной формой представления объектов являются окна: в окнах раскрываются папки, запускаются приложения, производится работа с документами и т.д.

Существуют два основных типа окон - окна приложений и окна документов. Окна приложений содержат приложения или папки. Окна документов содержат документы и открываются только внутри окон приложений.

Технология «**Plug and Play**» (включи и играй) позволяет автоматизировать подключение к компьютеру новых устройств и обеспечивает их конфигурирование. После включения компьютера операционная система определяет тип и конкретную модель установленного устройства и загружает необходимые для их функционирования драйверы.

Принципы внедрения и связывания объектов. Операционная система Windows позволяет:

1. Создавать комплексные документы, содержащие несколько разных типов данных.
2. Обеспечивать совместную работу нескольких приложений при подготовке одного документа.
3. Переносить и копировать объекты между приложениями.

Например, рисунок, созданный в графическом редакторе Paint, можно скопировать в текстовый документ, разрабатываемый в текстовом редакторе Microsoft Word. Возможность использования в одном документе объектов различной природы посредством внедрения и связывания является очень мощным инструментом Windows.

Под внедрением объектов подразумевается создание комплексного документа, содержащего два или более автономных объектов. Обычным средством внедрения объектов в документ является их импорт из готового файла, в котором данный объект хранится. Выполняется с помощью команды (Вставка – Объект). При сохранении комплексного документа происходит сохранение всех внедренных в него объектов. При этом размер исходного документа возрастает на величину внедренных в него объектов.

Связывание отличается от внедрения тем, что сам объект не вставляется в документ, а вместо этого вставляется только указатель на месторасположение объекта. Размер комплексного результирующего документа не увеличивается. Если документ готовится для печати на принтере или для просмотра на экране, то объекты в него вставляются методом связывания, если документ готовится для передачи во внешние структуры, то объекты в него внедряются. Приложения, которые способны создавать объекты для передачи другим приложениям называются OLE – серверами, а те которые позволяют внедрить или связывать чужие объекты в свои документы называются OLE – клиентами.

Служебные приложения Windows предназначены для обслуживания персонального компьютера и самой операционной системы. Они позволяют находить и устранять дефекты файловой системы, оптимизировать настройки программного и аппаратного обеспечения, а также автоматизировать некоторые рутинные операции, связанные с обслуживанием компьютера. К ним относятся:

1. Архивация данных - предназначена для автоматизации регулярного резервного копирования наиболее ценных данных на внешние носители (магнитные и оптические диски).
2. Буфер обмена - предназначен для просмотра текущего содержания буфера обмена Windows.
3. Дефрагментация диска – приложение, предназначенное для повышения эффективности работы жесткого диска путем устранения фрагментированности файловой структуры.
4. Проверка диска - позволяет выявлять логические ошибки в файловой структуре, а также физические ошибки, связанные с дефектами поверхности жесткого диска. Стандартную проверку рекомендуется проводить после каждого сбоя в работе компьютера, особенно после некорректного завершения работы с операционной системой. Полную проверку достаточно проводить два раза в год.
5. Сведения о системе – это специальный пакет программных средств, собирающих сведения о настройке операционной системы, ее приложений и оборудования компьютерной системы.
6. Системный монитор – программа, предназначенная для визуального или протокольного наблюдения за функционированием компьютера и операционной системы. Результаты наблюдения можно записывать в протокольный файл или отображать на экране в виде графиков.
7. Таблица символов - позволяет увидеть на экране все символы заданного набора и установить, какой символ какой клавише соответствует.
8. Очистка диска - может найти и освободить место, занимаемое файлами следующих категорий:

- временные файлы Интернета – файлы, которые копируются из Интернета для быстрого

Работа со сжатыми данными. Хранение и передача информации обходится недешево, поэтому регулярно возникает необходимость сжимать данные, перед тем как размещать их в архивах или передавать по каналам связи. Соответственно, существует и обратная необходимость восстановления данных из предварительно уплотненных архивов.

Характерной особенностью большинства типов данных является определенная избыточность. Степень избыточности зависит от типа данных. Наиболее избыточны видеоданные, затем идут графические, далее текстовые. Для человека избыточность данных связана с представлением о качестве, так как избыточность улучшает восприятие.

Сжатие информации – это процесс преобразования информации, хранящейся в файле, к виду, при котором уменьшается избыточность в ее представлении и соответственно требуется меньший объем для хранения. Для документов применяют аналогичное понятие архивация данных.

Программы, выполняющие операции со сжатыми данными называются архиваторами.

В зависимости от того, в каком объекте размещены данные, подвергаемые сжатию различают:

- уплотнение (архивацию) файлов;
- уплотнение (архивацию) папок;
- уплотнение дисков.

Архивация файлов и папок – помещение (загрузка) исходных файлов и папок в архивный файл в сжатом или несжатом виде. Разархивация - процесс восстановления файлов из архива.

Уплотнение файлов применяют для уменьшения их размеров при подготовке к передаче по каналам электронных сетей или к транспортировке на внешнем носителе малой емкости, например, на гибком диске. Уплотнение папок используют как средство архивации данных перед длительным хранением. Уплотнение дисков служит целям повышения эффективности использования их рабочего пространства и как правило применяется к дискам. имеющим недостаточную емкость.

Теоретически существуют три способа уменьшения избыточности данных – а) изменение содержания данных, б) изменение их структуры, в) сочетание первого и второго.

Степень сжатия файлов характеризуется коэффициентом сжатия:

$$Kc = (Vc / Vo) * 100\ \%\ ,$$

где Vc – объем сжатого файла; Vo – объем исходного файла.

Методы сжатия с регулируемой потерей информации - при сжатии данных происходит изменение их содержания, метод сжатия необратим, при разархивировании не происходит полного восстановления исходной информации. Они применимы для тех типов данных, для которых формальная утрата части содержания не приводит к значительному снижению потребительских свойств (некоторые аудио- и видеоданные, рисунки: .mp3, .mpg, .jpg). Методы сжатия с потерей информации обычно обеспечивают наиболее высокую степень сжатия, но их применимость ограничена в отношении текстовых документов, баз данных и программ.

Обратимые методы сжатия данных - при сжатии данных происходит только изменение их структуры. Обратимые методы применяются для любых типов данных. Характерными форматами сжатия без потери информации являются файлы с расширениями: .gif, .tif, .pcx – для графических данных; .avi – для видеоданных; .lzh, .lh, .cab – для любых типов данных. Классическими форматами сжатия любых типов данных являются широко используемые в работе компьютера форматы: .zip, .arj, .rar.

В случаях, когда архивация производится для передачи документа потребителю, следует предусмотреть наличие у него программного средства, необходимого для извлечения исходных данных из уплотненного архива. Если таких средств у пользователя нет, создают самораспаковывающиеся архивы. Сам архив получает расширение .exe, характерное для исполняемых файлов. Потребитель сможет выполнить его запуск как программы, после чего распаковка архива произойдет на его компьютере автоматически.

Большие по объему архивные файлы могут быть размещены на нескольких дисках или томах. Такие архивы называются многотомными. Том – это составная часть многотомного архива. Создавая архив из нескольких частей, можно записать его части на несколько дискет.

ЭТАПЫ РЕШЕНИЯ ЗАДАЧИ НА ЭВМ

Логически непротиворечивая последовательность этапов решения задачи на компьютере образует универсальную технологическую цепочку, позволяющую достигать поставленной цели. Часть этапов выполняется (может выполняться) без использования компьютера.

Этап 1. Анализ и исследование задачи подразумевает осуществление постановки задачи и четкое определение того, что дано, и что требуется найти.

Так, если задача конкретная, то под постановкой задачи понимают ответ на два вопроса: какие исходные данные известны и что требуется определить.

Если задача обобщенная, то при постановке задачи понадобится еще ответ на третий вопрос: какие данные допустимы.

Выполнение этапа возможно без использования компьютера.

Анализ и исследование задачи:

- сбор информации о задаче
- формулировка условия задачи
- описание данных: их типов, диапазонов величин, структуры и др.
- определение конечных целей решения, моделирования задачи
- определение формы выдачи результатов

Этап 2. Формализация задачи, построение модели. Содержанием этого этапа является построение математической модели, проведение математической формализации поставленной задачи как системы математических соотношений - формул, уравнений, неравенств и т. д., отражающих существенные свойства объекта или явления.

Необходимо отметить, что при построении математических моделей далеко не всегда удается найти формулы, явно выражающие искомые величины через данные. В таких случаях используются математические методы, позволяющие дать ответы той или иной степени точности.

В случае большого числа параметров, ограничений, возможных вариантов исходных данных модель явления может иметь очень сложное математическое описание (правда, реальное явление еще более сложно), поэтому часто построение математической модели требует упрощения требований задачи.

Необходимо выявить самые существенные свойства объекта, явления или процесса, закономерности; внутренние связи, роль отдельных характеристик. Выделив наиболее важные факторы, можно пренебречь менее существенными.

Выполнение этапа возможно без использования компьютера.

Формализация задачи:	**Построение модели:**
• выделить предположения, на которых будет основываться математическая модель; • определить, что считать исходными данными и результатами; • записать математические соотношения, связывающие результаты с исходными данными.	• анализ существующих аналогов; • анализ технических и программных средств; • разработка математической модели; • разработка структур данных.

Этап 3. Разработка алгоритма построения компьютерной модели. Наиболее эффективно математическую модель можно реализовать на компьютере в виде алгоритмической модели.

Для этого может быть использован язык блок-схем или, например учебный алгоритмический язык.

Выполнение этапа возможно без использования компьютера.

Этап 4. Программирование, компьютерный эксперимент. Содержание этого этапа – собственно программирование на определенном языке в определенной системе программирования.

В приведенном ниже перечне выполняемых работ в скобках указаны варианты, когда при решении некоторых задач можно обойтись без составления программы на языке программирования: это можно успешно сделать, используя современные пользовательские приложения – электронные таблицы, системы управления базами

3. Разработка алгоритма:

- выбор метода проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы или др.);
- выбор тестов и метода тестирования;
- проектирование самого алгоритма.

4. Программирование, компьютерный эксперимент:

- выбор языка программирования (или программного обеспечения);
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования (или последовательности технологических приемов в различных прикладных средах)

Этап 5. Тестирование и отладка программы. Под отладкой программы понимается процесс испытания работы программы и исправления выявленных при этом ошибок.

Обнаружить ошибки, связанные с нарушением правил записи программы на языке программирования (синтаксические и семантические ошибки), помогает используемая система программирования. Пользователь получает сообщение об ошибке, исправляет ее и снова повторяет попытку исполнить программу.

Проверка на компьютере правильности алгоритма производится с помощью тестов. Тест - это конкретный вариант значений исходных данных, для которого известен ожидаемый результат. Прохождение теста - необходимое условие правильности программы. На тестах проверяется правильность реализации программой запланированного сценария.

Тестирование и отладка программы:

- синтаксическая отладка
- отладка семантики и логической структуры программы
- тестовые расчеты и анализ результатов тестирования
- совершенствование программы

Этап 6. Проведение расчетов, анализ полученных результатов: происходит использование уже разработанной программы для получения искомых результатов. Производится анализ результатов решения задачи и, в случае необходимости, - уточнение математической модели с последующей корректировкой алгоритма и программы.

Таким образом, если это требуется, осуществляется «откат» на более ранние этапы технологической цепочки решения задач на ЭВМ: с повторным выполнением этапов 2 - 5.

Развитие программы. Программы, имеющие большое практическое или научное значение, используются длительное время. В процессе эксплуатации программы могут исправляться, дорабатываться – это и называется сопровождением, поддержкой программы.

Развитие программы:

- доработка программы для решения конкретных задач (типа задач)
- составление документации к решенной задаче, к математической модели, к алгоритму, программе, набору тестов, к использованию
- устранение ошибок, выявленных в ходе эксплуатации программы
- совершенствование программы, расширение сферы ее использования
- тиражирование программы, выпуск обновленных версий
- прекращение эксплуатации программы, ее утилизация

Задача

Какой математический аппарат необходим? Будем решать её на компьютере? С помощью какой программы?

Два человека стоят на расстоянии 10 метров друга от друга. У одного из них на носу сидит муха.

Одновременно и с одинаковой скоростью люди начинают двигаться друг навстречу другу. Каждый из них за каждые 10 секунд преодолевает 1 метр расстояния.

В момент начала движения людей муха тоже начинает двигаться: она непрерывно перелетает с носа одного человека на нос другого человека со скоростью 1 метр в секунду.

Какое расстояние пролетит муха к тому моменту, когда носы людей соприкоснутся?

МОДЕЛИРОВАНИЕ

Моделирование – это метод познания, состоящий в изучении реального объекта посредством создания и исследования его модели.

Под моделированием понимается процесс исследования реальной системы, включающий в себя:

- построение модели,
- изучение ее свойств,
- перенос полученных сведений на исследуемый объект.

Модель - это объект, который имеет сходство в некоторых отношениях с прототипом и служит средством описания и/или объяснения, и/или прогнозирования поведения прототипа.

Модель отражает наиболее существенные для исследования свойства изучаемого объекта. Один и тот же объект может иметь множество моделей, а разные объекты могут описываться одной моделью.

Для классификации моделей используется множество разнообразных признаков, в соответствии с которыми различают статические и динамические модели, аналитические и информационные модели, предметные и образно-знаковые модели, масштабные и немасштабные и т. д.

Каждый признак дает определенное знание о свойствах и модели, и моделируемой реальности. Признак может служить подсказкой о способе выполненного или предстоящего моделирования.

Дискретность и непрерывность.

Дискретность - характерный признак именно компьютерных моделей. Ведь компьютер может находиться в конечном, хотя и очень большом количестве состояний. Поэтому даже если

Случайность и детерминированность.

Неопределенность, случайность изначально противостоит компьютерному миру: ведь запущенный вновь алгоритм должен повториться и дать те же результаты. Но при этом именно компьютер используется для имитации случайных процессов, в которых используются генераторы случайных чисел.

Введение случайности в детерминированные задачи приводит к мощным и интересным моделям (например, вычисление площади методом случайных бросаний).

Матричность и скалярность.

Наличие у модели параметров матричности говорит о ее большей сложности и, возможно, большей точности по сравнению со скалярной. Например, если не выделить в населении страны все возрастные группы, рассматривая его изменение как целое, получим скалярную модель (например, модель Мальтуса), а если выделить эти группы - матричную (половозрастную). Именно матричная модель позволила объяснить колебания рождаемости после войны.

Статичность и динамичность.

Эти свойства модели обычно предопределяются свойствами реального объекта. Здесь нет свободы выбора. Просто статическая модель может быть шагом к динамической, либо часть переменных модели может считаться пока неизменной. Например, спутник движется вокруг Земли, на его движение влияет Луна. Если считать Луну неподвижной за время оборота спутника, получим более простую модель.

Аналитические модели.

Описание процессов аналитически, формулами и уравнениями. Но при попытке построить график удобнее иметь таблицы значений функции и аргументов.

Информационные модели.

Информационные модели принято противополагать математическим, точнее алгоритмическим. Здесь важно соотношение объемов данные/алгоритмы. Если данных больше или они важнее, то имеем информационную модель, иначе - математическую.

Предметные модели - это детская игрушка или глобус, например.

Образно-знаковые модели - это, прежде всего, модель в уме человека: образная, если преобладают графические образы, и знаковая, если больше слов и/или чисел. Образно-знаковые модели строятся на компьютере.

Масштабные модели - к ним относят те из предметных или образных моделей, которые повторяют форму объекта (например, географическая карта).

Классификация моделей по области использования:

- учебные модели – используются при обучении (наглядные пособия, тренажеры, обучающие программы);
- опытные – это уменьшенные или увеличенные копии изучаемого (проектируемого) объекта, которые используют для исследования и прогнозирования его будущих характеристик (автомобиль в аэродинамической трубе);
- научно-технические - создаются для исследования процессов и явлений (синхрофазотрон, стенд для проверки электронной аппаратуры, большой адронный коллайдер);
- игровые – репетиция поведения объекта в различных условиях (экономические, спортивные, деловые игры);
- имитационные – эти модели не просто отражают реальность в той или иной степени, а именно имитируют ее, эту реальность (испытание нового лекарства на лабораторных мышах).

Имитационный метод моделирования называется еще методом проб и ошибок. Имитационные модели появились давно в виде масштабных копий кораблей, мостов и др., но в связи с компьютерами рассматриваются недавно. Зная, как связаны элементы модели аналитически и логически, проще не решать систему неких соотношений и уравнений, а отобразить реальную систему в память компьютера, с учетом связей между элементами

Классификация моделей по фактору времени:

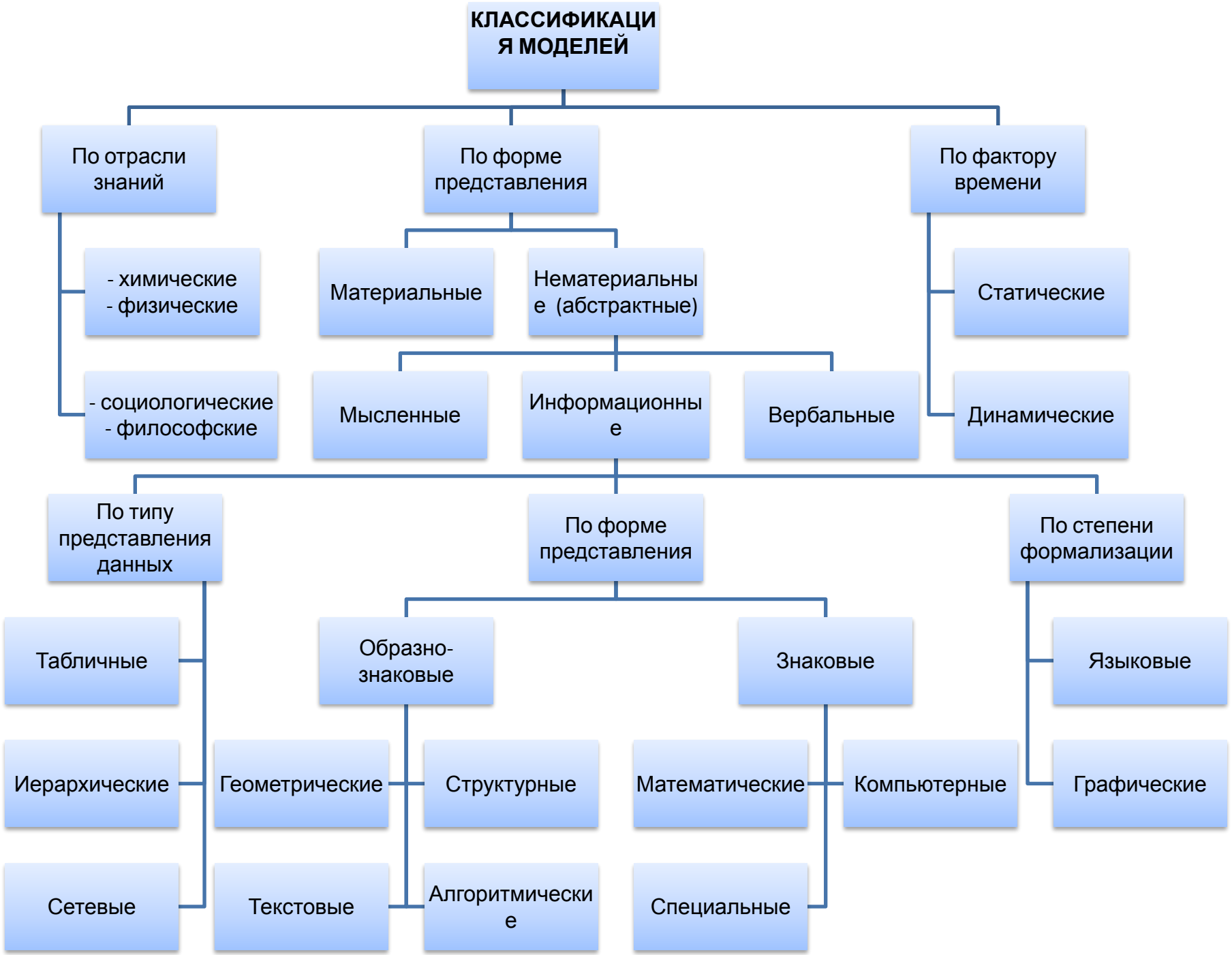
- статические – модели, описывающие состояние системы в определенный момент времени, дающие одномоментный срез информации по данному объекту (модель солнечной системы или планеты (глобус), классификация животных, строение молекулы, список (план) посаженных деревьев, отчет об обследовании состояния зубов в школе, описание конкретного автомобиля в определенный момент времени);
- динамические – модели, описывающие процессы изменения и развития системы, т. е изменения объекта во времени (описание движения тел, развития организмов, процесс химических реакций, а также карточка пациента (анамнез) в поликлинике).

Классификация моделей по отрасли знаний (деятельности человека):

- математические,
- биологические,
- химические,
- физические,
- геологические,
- социологические,
- исторические,
- политологические,
- философские,
- культурологические,
- экономические,
- управленческие,
- другие.

Классификация моделей по форме представления:

- материальные – это предметные (физические) модели, которые воспроизводят геометрические, физические и другие свойства объекта-оригинала в материальной форме. Они всегда имеют реальное воплощение. Отражают внешние свойства и внутреннее устройство исходных объектов, суть процессов и явлений исследуемого объекта. Такое моделирование используется в экспериментальном методе познания окружающей среды (анатомический муляж скелета человека, чучело, макет солнечной системы, школьные пособия (глобус), детская игрушка, лабораторная установка, физические и химические опыты);
- нематериальные (абстрактные) – не имеют реального воплощения, их нельзя потрогать или увидеть. Это объекты и процессы в абстрактной форме, их основу составляет


КЛАССИФИКАЦИЯ МОДЕЛЕЙ
По отрасли знаний
По форме представления
По фактору времени
- химические
- физические
- социологические
- философские
Материальные
Нематериальные (абстрактные)
Статические
Динамические
Мысленные
Информационные
Вербальные
По типу представления данных
По форме представления
По степени формализации
Табличные
Иерархические
Сетевые
Образно-знаковые
Геометрические
Структурные
Текстовые
Алгоритмические
Знаковые
Математические
Компьютерные
Специальные
Языковые
Графические

Абстрактные модели – это очень обширная группа моделей, которую, в свою очередь, также можно разбить на классы по различным основаниям (признакам). В частности, по признаку реализации:

- мысленные модели формируются в воображении человека в результате раздумий, умозаключений, а также - в виде некоторых образов. Такое моделирование всегда сопутствует сознательной деятельности человека;
- вербальные – те же мысленные модели, но выраженные в разговорной форме. Это моделирование используется для передачи мыслей, обмена образами.
- информационные модели – целенаправленно отобранная информация об объекте, которая отражает наиболее существенные для исследователя свойства этого объекта. Это совокупность информации, характеризующая свойства и состояния объекта, процесса, явления, а также его взаимосвязи с внешним миром.

Информационные модели, в свою очередь, подразделяются на:

а) по типу представления данных:

- табличные – объекты и их свойства представлены в виде списка, а их значения размещаются в ячейках прямоугольной формы. Перечень однотипных объектов размещен в первом столбце (или строке), а значения их свойств размещаются в следующих столбцах (или строках);
- иерархические – объекты распределены по уровням. Каждый элемент высокого уровня состоит из элементов нижнего уровня, а элемент нижнего уровня может входить в состав только одного элемента более высокого уровня (различного рода классификации, генеалогическое древо, организационно-штатные и функциональные структуры управления);
- сетевые – применяются для отражения систем, в которых связи между элементами имеют сложную структуру (сеть Интернет, транспортная сеть, электросеть).

б) по степени формализации:

- языковые - используют естественные языки и формальные языки (системы счисления, языки программирования, язык математических формул). Например, описание структуры компьютера на естественном языке – языковая модель. Процесс построения информационных моделей с помощью формальных языков называется формализацией. Таким образом, наиболее глубоко формализованными моделями, построенными с использованием математических понятий и формул, являются математические модели (алгоритмы, формулы, системы уравнений и т. п.);
- графические - представляют собой рисунки, карты, схемы, чертежи, графики, диаграммы (схема компьютера).

в) по форме представления:

- образно-знаковые модели
  - геометрические (рисунок, пиктограмма, чертеж, карта, план, объемное изображение);
  - структурные (таблица, граф, схема, диаграмма);
  - текстовые (описание естественными языками) – эти модели используют последовательность предложений на формализованных диалектах естественного языка для описания той или иной области действительности (примерами такого рода моделей является милицейский протокол, правила дорожного движения);
  - алгоритмические (нумерованный список, пошаговое перечисление, блок-схема);
- знаковые модели:
  - математические – выражающие существенные черты объекта или процесса языком уравнений, математическими формулами, отображающими связь параметров, и других математических средств. Они традиционны для теоретической физики, механики, химии, биологии и ряда других, в том числе гуманитарных и социальных наук;
  - компьютерные – программы на языках программирования (фортран, алгол, бейсик);

ПОСТРОЕНИЕ МОДЕЛИ

КЛАССИФИКАЦИЯ ВИДОВ МОДЕЛИРОВАНИЯ СИСТЕМ

1) В зависимости от режима функционирования различаются:

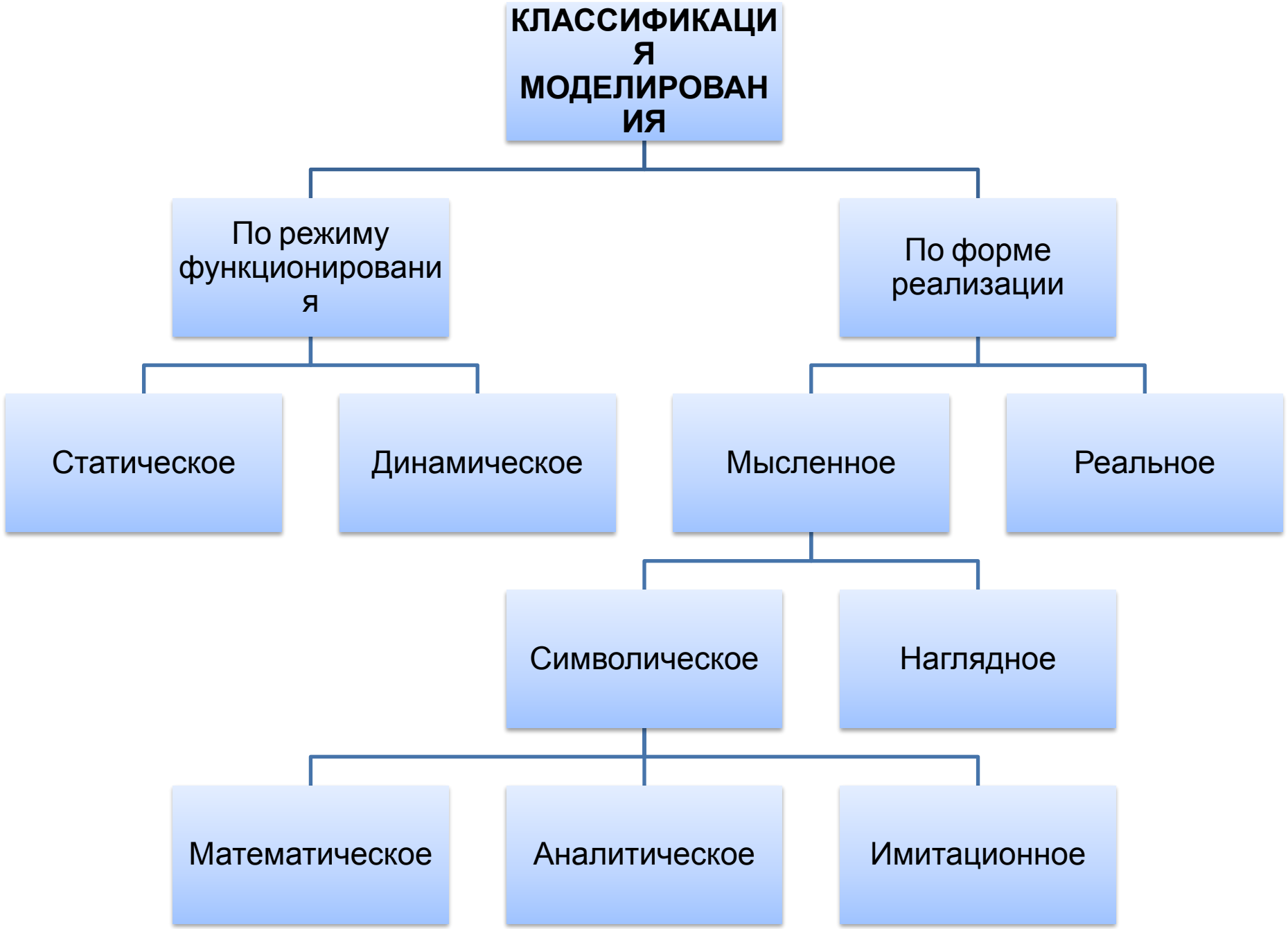
- **статическое** моделирование - служит для описания состояния объекта в фиксированный момент времени;
- **динамическое** - для исследования объекта во времени.

2) В зависимости от формы реализации:

- **реальное** моделирование - используется возможность исследования характеристик либо на реальном объекте целиком, либо на его части. Такие исследования проводятся как на объектах, работающих в нормальных режимах, так и при организации специальных режимов для оценки интересующих исследователя характеристик (при других значениях переменных и параметров, в другом масштабе времени и т. д.). Реальное моделирование является наиболее адекватным, но его возможности ограничены;
- **мысленное** моделирование - применяется тогда, когда модели не реализуемы в заданном интервале времени либо отсутствуют условия для их физического создания (например, ситуация микромира).

Мысленное моделирование реальных систем реализуется в виде наглядного и символическое. Для представления функциональных, информационных и событийных моделей этого вида моделирования разработано значительное количество средств и методов.

При **наглядном** моделировании на базе представлений человека о реальных объектах создаются наглядные модели, отображающие явления и процессы, протекающие в объекте. Примером таких моделей являются учебные плакаты, рисунки, схемы, диаграммы. Разновидностью наглядного моделирования является макетирование


КЛАССИФИКАЦИЯ МОДЕЛИРОВАНИЯ
По режиму функционирования
По форме реализации
Статическое
Динамическое
Мысленное
Реальное
Символическое
Наглядное
Математическое
Аналитическое
Имитационное

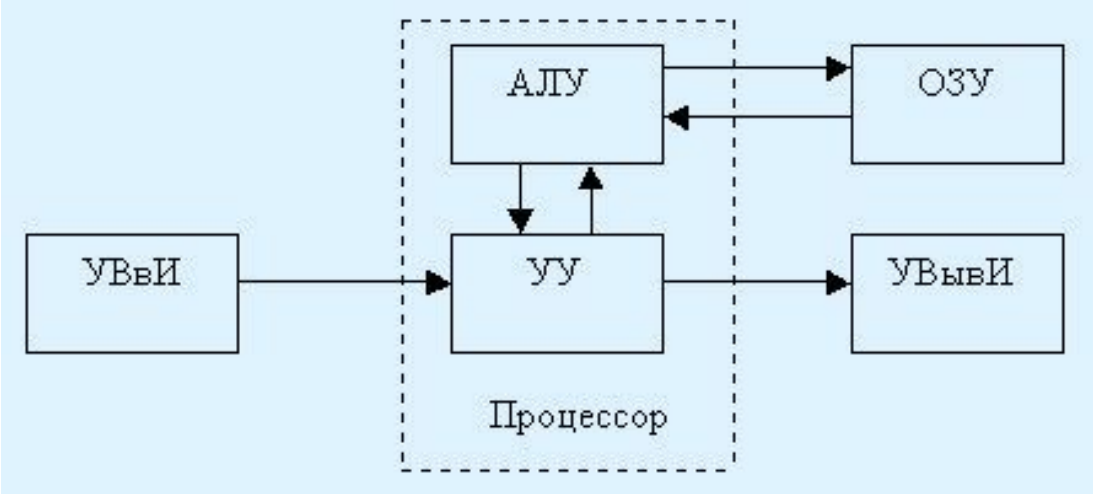
Разновидностями **символического** моделирования являются математическое, аналитическое, имитационное и другие виды моделирования.

Математическое моделирование - это процесс установления соответствия данному реальному объекту некоторого математического объекта, называемого математической моделью. Процесс построения математической модели называется **формализацией.**

Для **аналитического** моделирования характерно то, что в основном моделируется только функциональный аспект системы. При этом глобальные уравнения системы, описывающие закон (алгоритм) ее функционирования, записываются в виде некоторых аналитических соотношений или логических условий. Результат – функциональные модели.

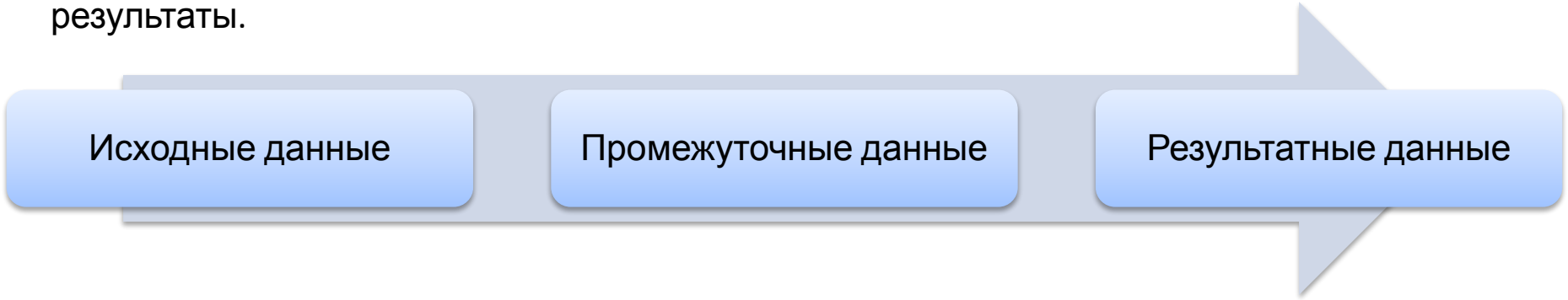
При **имитационном** моделировании воспроизводится алгоритм функционирования системы во времени - поведение системы, причем имитируются элементарные явления, составляющие процесс, с сохранением их логической структуры и последовательности протекания, что позволяет по исходным данным получить сведения о состояниях процесса в определенные моменты времени, дающие возможность оценить характеристики системы. Результат – событийные модели.

КАКАЯ ЭТО МОДЕЛЬ?



Величины

Совокупность величин, с которыми работает компьютер называется данные. По отношению к программе обработки данных величины делятся на исходные, промежуточные, результаты.



Три основных свойства величины:		
Имя величины	Значение величины	Тип величины

На уровне команд процессора величина идентифицируется при помощи адреса ячейки памяти в которых она хранится. В алгоритмах и языках программирования величины делятся на константы переменные.

Константа – неизменная величина и в алгоритме она представляется собственным значением, например, 15, 34, а, true и т. д.

Переменные величины могут изменять свои значения в ходе выполнения программы и представляются символическими именами – идентификаторами например, x, s2.

Любая константа и переменная занимает ячейку памяти. Типы констант определяются по форме записи в тексте, типы переменных устанавливаются в описаниях переменных.

По структуре данные делятся на простые и структурированные. Для **простых или скалярных величин** справедливо утверждение: одна величина – одно значение. Для **структурированных:** одна величина – множество значений. К структурированным величинам относятся массивы, строки, множества и т. д.

В каждом языке программирования существует своя концепция типов данных и своя система типов. Тем не менее, в любой язык программирования входит минимальный набор основных типов данных.

Тип величины	Множество допустимых значений	Множество допустимых операций	Форма внутреннего представления
Целый	Целые положительные и отрицательные числа в некотором диапазоне	Арифметические операции с целыми числами + - * , целое деление и остаток от деления; операции отношений < > = и др.	Формат с фиксированной точкой
Вещественный	Любые целые и дробные числа в некотором диапазоне	Арифметические операции с целыми числами + - * / ; операции отношений < > =	Формат с плавающей точкой
Логический	True (истина) False (ложь)	Логические операции И, ИЛИ, НЕ; операции отношений	1 бит 1 – true 0 – false
Символьный	Любые символы компьютерного алфавита: a, 5, +, $	Операции отношений	Коды таблицы символьной кодировки, 1 символ равен 1 байту

Понятие алгоритма

Алгоритм – это последовательность точных предписаний (команд), понятных исполнителю, совершить конечную последовательность действий, направленных на достижение конкретного результата.

Другими словами, это точно определенное описание способа решения задачи в виде конечной по времени последовательности действий. Такое описание называется формальным.

Исполнитель – устройство или живое существо, которое выполняет по определенным правилам составленный алгоритм. Исполнителя, который может и не понимать цели алгоритма, называют формальным исполнителем.

Команда – это указание исполнителю совершить некоторое действие. Набор всех команд исполнителя называется системой команд. Независимо от того, на каком языке программирования будет написана программа, алгоритм решения любой задачи на ЭВМ

Команды алгоритма

Команда ввода	Команда присваивания	Команда обращения к вспомога-тельному алгоритму	Команда ветвления	Команда цикла	Команда вывода

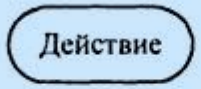
Свойства алгоритма:

1. Последовательность алгоритма.
2. Определенность алгоритма. Каждая команда алгоритма (предписание, выдаваемое на каждом шаге) должна быть понятна исполнителю, не должна оставлять места для неопределенного или неоднозначного исполнения им этих шагов.
3. Дискретность алгоритма. Процесс решения задачи, определяемый алгоритмом, расчленен на отдельные элементарные шаги, и алгоритм представляет собой последовательность указаний, команд, определяющих порядок выполнения шагов процесса.
4. Конечность алгоритма.
5. Результативность алгоритма. Алгоритм всегда должен приводить не важно, через какое количество шагов, большее или меньшее, лучше меньшее к результату.
6. Эффективность алгоритма.
7. Массовость алгоритма. Алгоритм, разработанный для определенного типа задач, должен быть применим для решения этих задач при любых допустимых исходных данных.

Формы представления (записи) алгоритмов:

1. Вербальная форма – используется для алгоритмов, ориентированных на исполнителя-человека (например, должностная инструкция).
2. Программа - алгоритм, записанный на понятном компьютеру языке программирования.
3. Блок-схема - представляет алгоритм в наглядной графической форме. Команды алгоритма помещаются внутрь блоков, соединенных стрелками, показывающими очередность выполнения команд алгоритма.

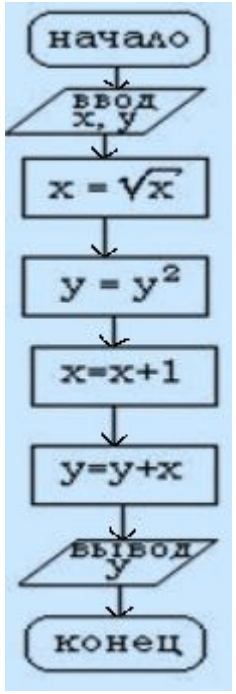
Некоторые элементы блок-схемы

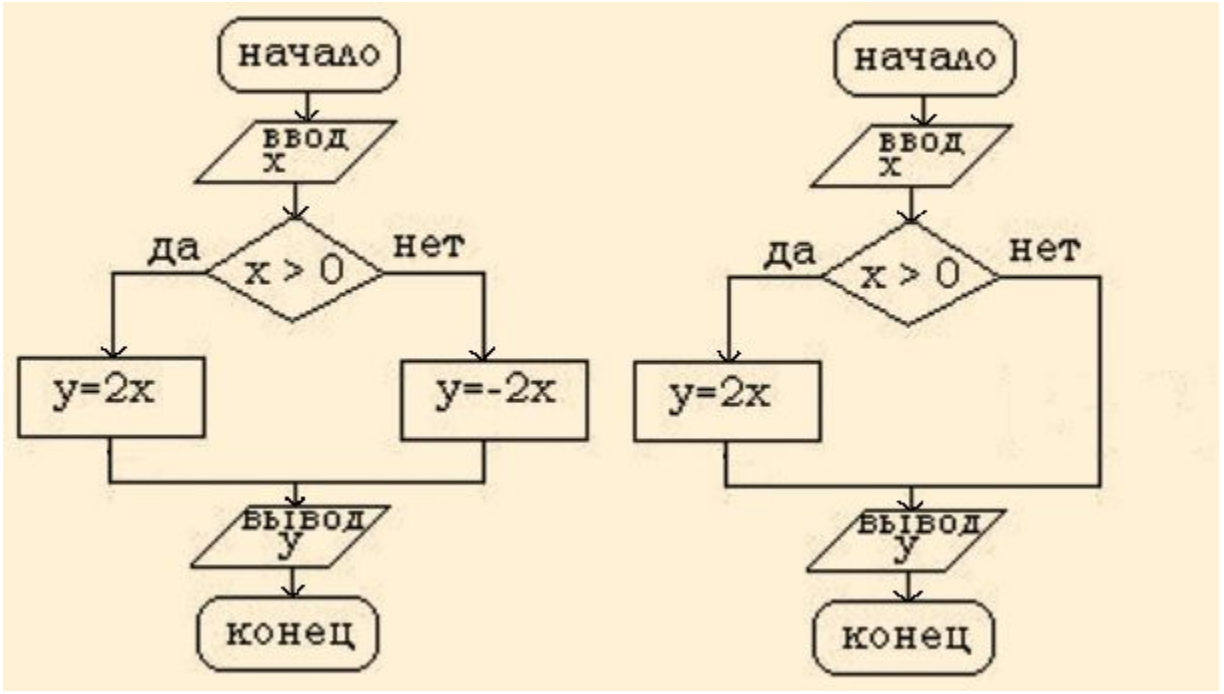
Название блока	Обозначение	Назначение блока
Терминатор		Начало, завершение программы или подпрограммы
Процесс		Обработка данных (вычисления, пересылки и т. п.)
Данные		Операции ввода-вывода
Решение		Ветвления, выбор, итерационные и поисковые циклы
Подготовка		Счетные циклы
Граница цикла		Любые циклы
Предопределенный процесс		Вызов процедур
Соединитель		Маркировка разрывов линий
Комментарий		Пояснения к операциям

Основные виды алгоритмов

1. Линейный алгоритм. Все действия, команды (записанные в блоки) выполняются последовательно, одна за другой. Линейный алгоритм состоит из команд ввода, присваивания, обращения к вспомогательным алгоритмам, вывода.

2. Алгоритм ветвления. В этом всегда имеется блок условия. В этой форме организации действий в зависимости от выполнения или невыполнения конкретного условия, совершается либо одна, либо другая последовательность действий. Различаются <u>две формы ветвления: полная и неполная</u>.





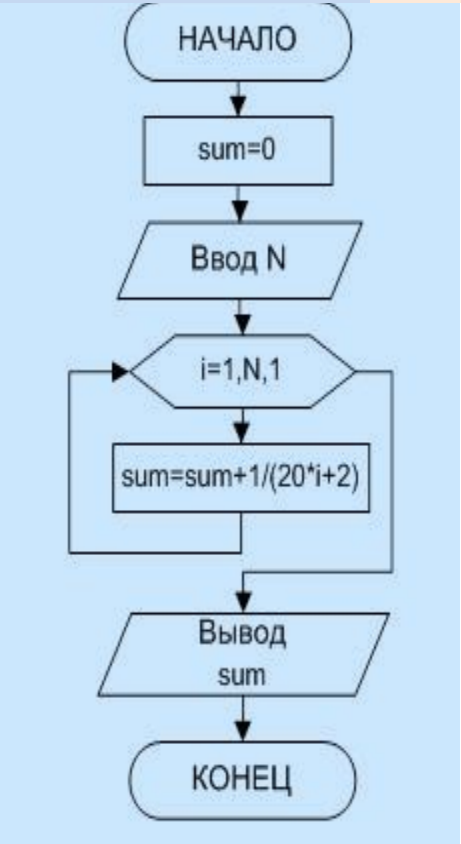
3. Циклические алгоритмы - это такая форма организации действий, при которой одна и та же последовательность действий повторяется несколько раз (или ни разу) до тех пор, пока выполняется некоторое условие. **Цикл** - совокупность многократно выполняющихся блоков

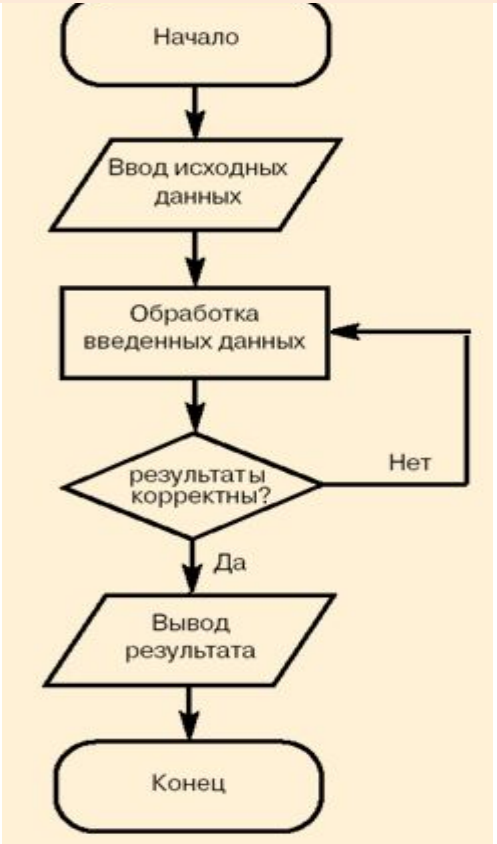
Циклы со счетчиком: тело цикла выполняется определенное количество раз (цикл «для»)

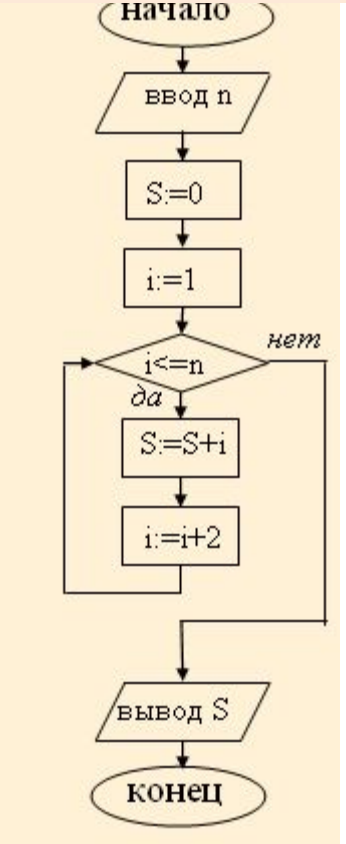
Циклы с условием: тело цикла выполняется до тех пор, пока выполняется условие

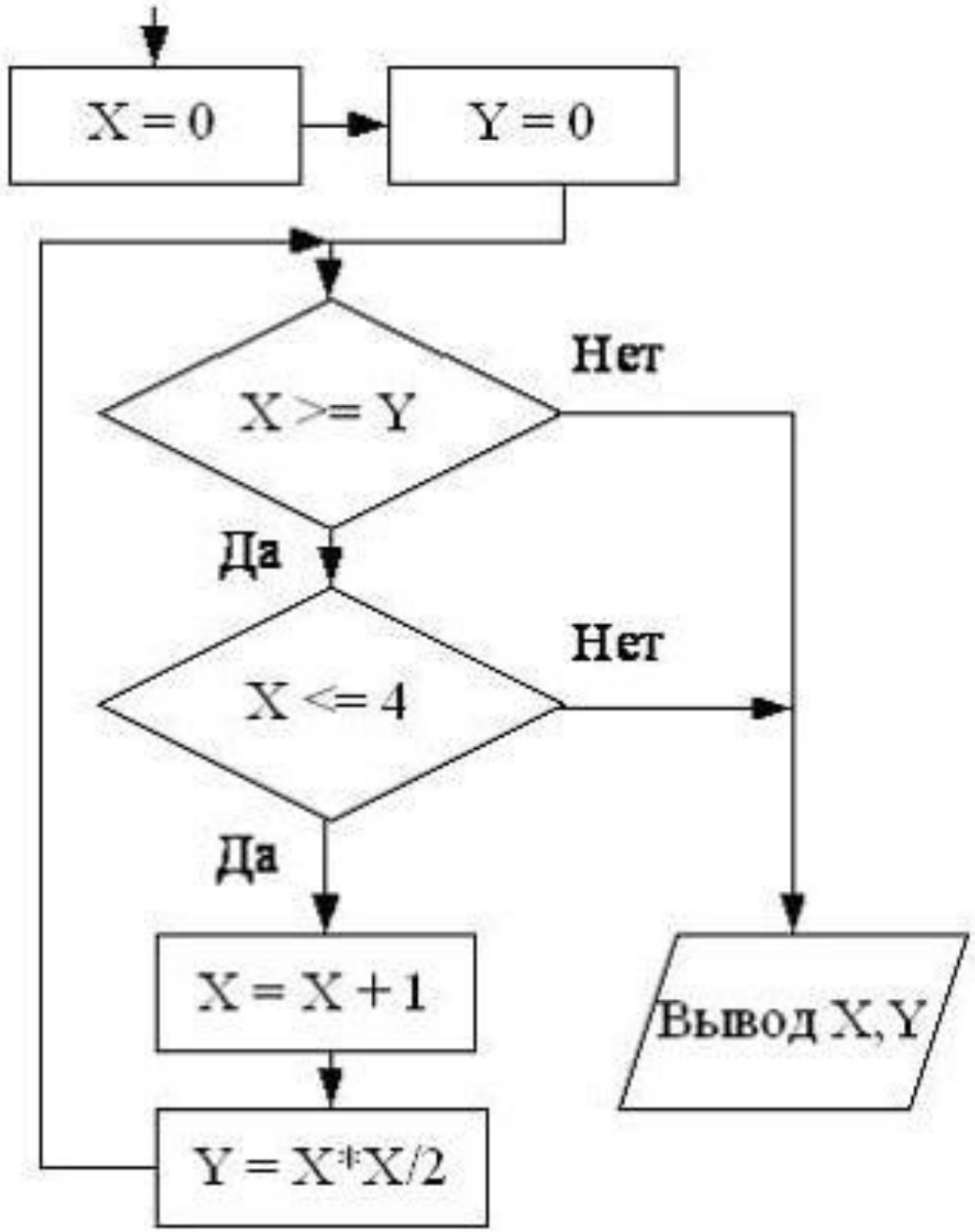
- <u>циклы с предусловием</u>: выполнение тела цикла повторяется, пока условие цикла истинно. Когда условие цикла становится ложным, цикл заканчивает выполнение (цикл «пока»)

- <u>циклы с постусловием</u>: используется условие окончания цикла. Когда оно становится истинным, цикл заканчивает работу (цикл «до»)








X = 0
Y = 0
X >= Y
Нет
Да
X <= 4
Нет
Да
X = X + 1
Y = X*X/2
Вывод X, Y

Этот сложный русский язык

И дико мне – иди ко мне.
Покалечилась – пока лечилась.
Ты жеребенок – ты же ребенок.
Ему же надо будет – ему жена добудет.
Несуразные вещи – несу разные вещи.
Промежуток – промеж уток.
Надо ждать – надо ж дать.
Помаши нам – по машинам.
Мы женаты – мы же на ты.
Задело – за дело.

ОСНОВЫ ПРОГРАММИРОВАНИЯ

Команды, поступающие в процессор компьютера по его шинам, являются электрическими сигналами, которые можно представить как совокупность нулей и единиц, т. е. числами. Разным командам соответствуют разные числа. Поэтому программа, с которой реально работает микропроцессор, называется **машинным кодом**.

Компьютер управляется по определенному алгоритму. Для представления алгоритма в виде, понятном компьютеру, служат языки программирования. Сначала всегда разрабатывается алгоритм действий, а потом он записывается на одном из таких языков. В итоге получается текст программы – полное, законченное и детальное описание алгоритма на языке программирования.

Затем специальными служебными приложениями, которые называются трансляторами (компиляторами) , этот текст программы переводится в машинный код или сразу исполняется в соответствии с командами языка, указанными в тексте программы, с помощью программ – интерпретаторов. В реальных системах программирования перемешаны технологии компиляции и интерпретации.

Язык программирования – искусственный язык. От естественных они отличаются ограниченным числом слов, значение которых понятно транслятору, и очень строгими правилами записи команд, которые называются операторами.

Совокупность подобных требований образует синтаксис языка программирования, а смысл каждой команды и других конструкций языка – его семантику. Нарушение формы записи программы приводит к тому, что транслятор не может понять назначение оператора и выдает сообщение о синтаксической ошибке, а правильно написанное, но не отвечающее алгоритму использования языка приводит к семантическим или логическим ошибкам. Процесс поиска ошибок в программе называется тестированием, процесс устранения ошибок – отладкой.

Разные типы процессоров имеют разные наборы команд.

Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется **языком программирования низкого уровня**.

зыком самого низкого уровня является язык ассемблера, который представляет собой каждую команду машинного кода с помощью символьных условных обозначений, называемых мнемониками.

С помощью языков низкого уровня создаются очень эффективные и компактные программы, так как разработчик получает доступ ко всем возможностям процессора. С другой стороны, требуется высокий уровень подготовки разработчика и программа не может быть перенесена на компьютер с другим типом процессора.

Язык программирования высокого уровня значительно ближе и понятнее человеку, чем компьютеру. Особенности конкретных компьютерных архитектур в них не учитываются.

Разрабатывать программы на языках высокого уровня с помощью понятных команд значительно проще, а ошибок при создании значительно меньше.

В самом общем случае для создания программы на выбранном языке программирования нужно иметь следующие компоненты:

1. Текстовый редактор.
2. Программа – компилятор.
3. Редактор связей.
4. Библиотека функций.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Из универсальных языков программирования сегодня наиболее распространены следующие:

1. Бэйсик – для освоения требует начальной подготовки, для общеобразовательных школ.
2. Паскаль – требует специальной подготовки для школ с углубленным изучением информатики и общетехнических вузов.
3. Си++, Джава – требуют профессиональной подготовки (специализированные средние и высшие учебные заведения).

В последние годы в программировании появился так называемый визуальный подход, особенно широко используемый в Windows. Все необходимые элементы оформления и управления создаются и обслуживаются не путем ручного программирования, а с помощью готовых визуальных компонентов, которые с помощью мыши перетаскиваются в проектируемое окно. Их свойства и поведение затем настраиваются с помощью редакторов, визуально показывающих характеристики соответствующих элементов. В результате программирование заменяется на проектирование.

Для каждого из этих языков имеются немало систем программирования. Наиболее популярны следующие визуальные среды быстрого проектирования программ для Windows:

1. Microsoft Visual Basic.
2. Borland Delphi
3. Borland C++Builder
4. Symantec Café.

Вообще языки программирования принято делить на пять поколений.

В **первое поколение** входят языки, созданные в начале 50-х годов, когда первые компьютеры только появились на свет. Это был первый язык ассемблера, созданный по принципу «одна инструкция — одна строка».

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

Появление **третьего поколения** языков программирования принято относить к 60-м годам. В это время родились универсальные языки высокого уровня, с их помощью удается решать задачи из любых областей. Такие качества новых языков, как относительная простота, независимость от конкретного компьютера и возможность использования мощных синтаксических конструкций, позволили резко повысить производительность труда программистов. Понятная большинству пользователей структура этих языков привлекла к написанию небольших программ (как правило, инженерного или экономического характера) значительное число специалистов из не компьютерных областей. Подавляющее большинство языков этого поколения применяется и сегодня.

С начала 70-х годов по настоящее время продолжается период языков **четвертого поколения**. Эти языки предназначены для реализации крупных проектов, повышения их надежности и скорости создания. Они обычно ориентированы на специализированные области применения, где хороших результатов можно добиться, используя не универсальные, а проблемно-ориентированные языки, оперирующие конкретными понятиями узкой предметной области. Как правило, в эти языки встраиваются мощные операторы, позволяющие одной строкой описать такую функциональность, для реализации которой на языках младших поколений потребовались бы тысячи строк исходного кода.

Рождение языков **пятого поколения** произошло в середине 90-х годов. К ним относятся также системы автоматического создания прикладных программ с помощью визуальных средств разработки, без знания программирования. Главная идея, которая закладывается в эти языки, — возможность автоматического формирования результирующего текста на универсальных языках программирования (который потом требуется откомпилировать). Инструкции же вводятся в компьютер в максимально наглядном виде с помощью методов, наиболее удобных для человека, не знакомого с программированием.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ

Fortran (Фортран). Это первый компилируемый язык, созданный Джимом Бэкусом в 50-е годы. Программисты, разрабатывавшие программы исключительно на ассемблере, выражали серьезное сомнение в возможности появления высокопроизводительного языка высокого уровня, поэтому основным критерием при разработке компиляторов Фортрана являлась эффективность исполняемого кода. Хотя в Фортране впервые был реализован ряд важнейших понятий программирования, удобство создания программ было принесено в жертву возможности получения эффективного машинного кода. Однако для этого языка было создано огромное количество библиотек, начиная от статистических комплексов и кончая пакетами управления спутниками, поэтому Фортран продолжает активно использоваться во многих организациях, постоянно ведутся работы над очередным, новым стандартом Фортрана. Имеется, например, стандартная версия Фортрана HPF (High Performance Fortran) – для параллельных суперкомпьютеров со множеством процессоров.

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

Algol (Алгол). Компилируемый язык, созданный в 1960 году. Он был призван заменить Фортран, но из-за более сложной структуры не получил широкого распространения. В 1968 году была создана версия Алгол 68, по своим возможностям и сегодня опережающая многие языки программирования, однако из-за отсутствия достаточно эффективных компьютеров для нее не удалось своевременно создать хорошие компиляторы.

Pascal (Паскаль). Язык Паскаль, созданный в конце 70-х годов основоположником множества идей современного программирования Никлаусом Виртом, во многом напоминает Алгол,

ужесточен ряд требований к структуре программы и имеются возможности, позволяющие успешно применять его при создании крупных проектов.

Basic (Бейсик). Для этого языка имеются и компиляторы, и интерпретаторы, а по популярности он занимает первое место в мире. Он создавался в 60-х годах в качестве учебного языка и очень прост в изучении.

С (Си). Данный язык был создан в лаборатории Bell и первоначально не рассматривался как массовый. Он планировался для замены ассемблера, чтобы иметь возможность создавать столь же эффективные и компактные программы, и в то же время не зависеть от конкретного типа процессора. Си во многом похож на Паскаль и имеет дополнительные средства для прямой работы с памятью. На этом языке написано множество прикладных и системных программ и ряд известных операционных систем (UNIX).

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

Джава, Ява. Этот язык был создан компанией Sun в начале 90-х годов на основе Си++. Он призван упростить разработку приложений на основе Си++ путем исключения из него всех низкоуровневых возможностей. Но главная особенность этого языка — компиляция не в машинный код, а в платформно-независимый байт-код (байтная команда занимает один байт). Этот байт-код может выполняться с интерпретатора — виртуальной Java-машины JVM (Java Virtual Machine), версии которой созданы сегодня для любых платформ. Благодаря наличию множества Java-машин программы на Java можно переносить не только на уровне исходных текстов, но и на уровне двоичного байт-кода, поэтому по популярности этот язык сегодня занимает второе место в мире после Бейсика.

Декларативные языки – Пролог (логический язык) и Лисп (функциональный язык). При использовании декларативного языка программист указывает исходные данные, взаимосвязи между ними, свойства результата. Алгоритмы не строятся.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ БАЗ ДАННЫХ

Эта группа языков отличается от алгоритмических языков прежде всего решаемыми задачами. База данных — это файл (или группа файлов), представляющий упорядоченный набор записей, имеющих единообразную структуру и организованных по единому шаблону (как правило, в табличном виде). База данных может состоять из нескольких таблиц. Удобно хранить в базах данных различные сведения из справочников, картотек, журналов бухгалтерского учета и т. д.

При работе с базами данных чаще всего требуется выполнять следующие операции:

- создание/модификация свойств/удаление таблиц в базе данных;
- поиск, отбор, сортировка информации по запросам пользователей;
- добавление новых записей;
- модификация существующих записей;
- удаление существующих записей.

Первые базы данных появились очень давно, как только появилась потребность в обработке больших массивов информации и выборки групп записей по определенным признакам. Для этого был создан структурированный язык запросов SQL (Structured Query Language). Он основан на мощной математической теории и позволяет выполнять эффективную обработку баз данных, манипулируя группами записей.

Для управления большими базами данных и их эффективной обработки разработаны СУБД (Системы Управления Базами Данных). Практически в каждой СУБД помимо поддержки языка SQL имеется также свой уникальный язык, ориентированный на особенности этой СУБД и не переносимый на другие системы. Сегодня в мире насчитывается пять ведущих производителей СУБД: Microsoft (SQL Server), IBM (DB2), Oracle, Software AG (Adabas), Informix и Sybase. Их продукты нацелены на поддержку одновременной работы тысяч пользователей в сети, а базы данных могут храниться в распределенном виде на нескольких серверах. В Oracle имеется встроенный язык PL/SQL, в Informix — INFORMIX 4GL, в Adabas — Natural и т. д.

С появлением персональных компьютеров были созданы так называемые настольные СУБД. Родоначальником современных языков программирования баз данных для ПК принято считать СУБД dBase II, язык которой был интерпретируемым. Затем для него были созданы компиляторы, появились СУБД FoxPro и Clipper, поддерживающие диалекты этого языка. Сегодня похожие, но несовместимые версии языков семейства dBase реализованы в продуктах Visual FoxPro фирмы Microsoft и Visual dBase фирмы Inprise.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ ДЛЯ ИНТЕРНЕТА

С активным развитием глобальной сети было создано немало реализаций популярных языков программирования, адаптированных специально для Интернета. Все они отличаются характерными особенностями: языки являются интерпретируемыми, интерпретаторы для них распространяются бесплатно, а сами программы — в исходных текстах. Такие языки называют скрипт-языками.

HTML. Общеизвестный язык для оформления документов. Он очень прост и содержит элементарные команды форматирования текста, добавления рисунков, задания шрифтов и цветов, организации ссылок и таблиц. Все Web-страницы написаны на языке HTML или используют его расширения.

Perl. В 80-х годах Ларри Уолл разработал язык Perl. Он задумывался как средство эффективной обработки больших текстовых файлов, генерации текстовых отчетов и управления задачами. По мощности Perl значительно превосходит языки типа Си. В него введено много часто используемых функций работы со строками, массивами, всевозможные средства преобразования данных, управления процессами, работы с системной информацией и др.

Tel/Tk. В конце 80-х годов Джон Аустираут придумал популярный скрипт-язык Tel и библиотеку Tk. В Tel он попытался воплотить видение идеального скрипт-языка. Tel ориентирован на автоматизацию рутинных процессов и состоит из мощных команд, предназначенных для работы с абстрактными нетипизированными объектами. Он независим от типа системы и при этом позволяет создавать программы с графическим интерфейсом.

VRML. В 1994 году был создан язык VRML для организации виртуальных трехмерных интерфейсов в Интернете. Он позволяет описывать в текстовом виде различные трехмерные сцены, освещение и тени, текстуры (покрытия объектов), создавать свои миры, путешествовать по ним, «облетать со всех сторон», вращать в любых направлениях, масштабировать, регулировать освещенность и т.д.

ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Выделяют три основных технологии программирования: алгоритмическое (модульное), структурное и
объектно-ориентированное.

Алгоритмическое программирование.

Основная идея алгоритмического программирования – разбиение программы на последовательность модулей, каждый из которых выполняет одно или несколько действий Выполнение модуля начинается с первой команды и заканчивается последней командой. В нем используются основные операторы (ввода, вывода, присваивания, цикла, ветвления). Применяется для решения несложных, небольших по объему данных задач.

Структурное программирование.

Применяется при создании средних по размеру приложений. Структура программы должна отражать структуру решаемой задачи, алгоритм решения должен быть ясно виден из исходного текста программы. Для этого в структурном программировании используются три основных вида алгоритмов (линейный, ветвления и циклический), а также подпрограммы.

Подпрограмма – это набор процедур (включая операторы, функции и команды), выполняющих нужное действие и не зависящих от других частей программы.

В данном случае программа разбивается на множество подпрограмм, каждая из которых выполняет одно из действий. Подпрограммы делятся на процедуры и функции. Процедура выполняет группу операторов, функция вычисляет некоторое значение. Возможность применения подпрограмм относит язык программирования к классу процедурных языков.

Структура программы, создаваемой по такой технологии, сверху вниз: основная программа – подпрограмма – процедура – функция – команда.

Объектно-ориентированное программирование.

Объект – совокупность свойств (структур данных, характерных для этого объекта), методов их обработки (подпрограмм изменения свойств) и событий, на которые данный объект может реагировать, и которые как правило приводят к изменению свойств.

Объединение в объекте его свойств и возможных над ним операций называется инкапсуляцией.

Примеры объектов - папки, документы, окна в Windows, в Word – фрагменты текста, символы, в Excel – листы, диаграммы, формулы, в Paint – графические примитивы (точка, линия, окружность, прямоугольник.

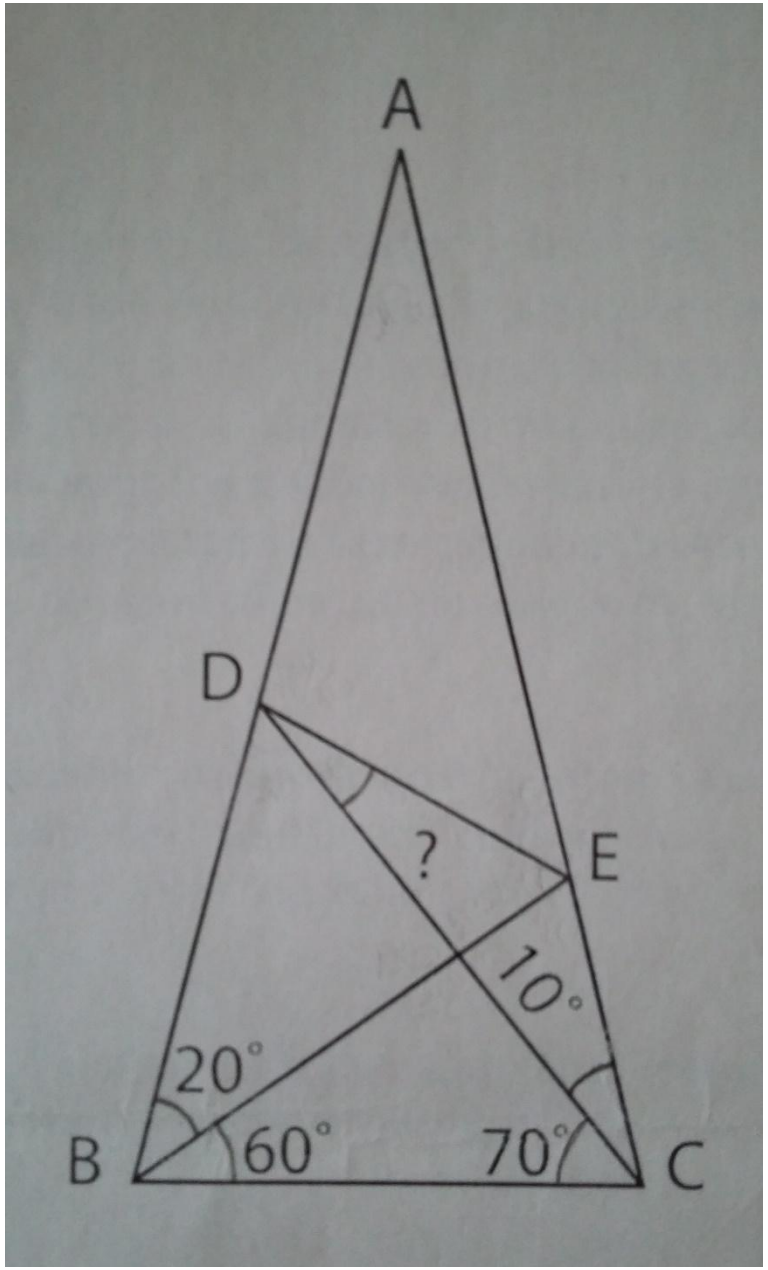
Объекты могут иметь идентичную структуру и отличаться только значениями свойств. В таком случае объекты образуют класс. Например, файлы разных типов образуют класс.

Конкретный объект, имеющий структуру этого класса, называется экземпляром.

Важнейшая характеристика класса – наследование. Это возможность создания на основе класса новых с наследованием всех свойств старого. В данном случае исходный класс называется классом – родителем, новый класс – классом – потомком. Класс, который не имеет предшественника, называется базовым.

Возможность проведения одних и тех же операций с разными объектами при сохранении разных методов их реализации называется полиморфизмом.

Сегодня известны 6 способов решения этой задачи, предложите хотя бы один



- Дан равнобедренный треугольник ABC, т.е. сторона AB равна стороне AC.
- Из вершины C к стороне AB проведена линия CD. Угол BCD составляет 70^{o}, а угол DCA = 10^{o}.
- Из вершины B к стороне AC проведена линия BE. Угол CBE составляет 60^{o}, а угол ABE = 20^{o}.
- Точки D и E тоже соединены линией.
- Найти величину угла CDE.
- *Схема приведена без соблюдения масштаба.*

Спасибо за внимание!