Тема 4.1.Алгоритмы и программирование задач

1. Основные этапы автоматизации решения задач на ЭВМ
2. Алгоритм, его свойства и способы представления
3. Разработка программ

1. Основные этапы автоматизации решения задач на ЭВМ

Информатика – наука, систематизирующая приемы и методы создания, хранения, воспроизведения, обработки и передачи данных (информации) средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ими.

Основная задача информатики – систематизация приемов и методов работы с аппаратными и программными средствами вычислительной техники.

Цель систематизации – выделение, внедрение, развитие передовых, наиболее эффективных технологий, в автоматизации этапов работы с данными, а также в методическом обеспечении новых технологических исследований.

Предметами информатики являются:
аппаратное обеспечение средств ВТ;
программное обеспечение средств ВТ;
средства взаимодействия аппаратного и программного обеспечения;
средства взаимодействия человека с аппаратными и программными средствами (пользовательский интерфейс).

Основные направления практического применения информатики

Разработка приемов и методов построения вычислительных систем.

Разработка приемов и методов управления аппаратным и программным обеспечением.

Разработка приемов, методов и средств разработки программ (программирование).

Защита информации в вычислительных системах.

Автоматизация функционирования программно-аппаратных средств.

Стандартизация аппаратных и программных средств, их интерфейсов, форматов представления данных.

Оценка эффективности функционирования вычислительных систем и программного обеспечения.

Характеристика процессов решения задач

Человек

Решает осмысленно и при случае может прекратить вычисления, обратиться за консультацией, повторить часть вычислений и т. д.

Человек имеет возможность интуитивного решения задач с учетом накопленного опыта.

ЭВМ

ЭВМ будет выполнять лишь то, что предписано, разработанной программой.

У ЭВМ полностью отсутствуют интуитивные возможности и поэтому весь процесс решения задач должен быть предварительно осознан человеком.

Этого можно добиться, если формализовать способ решения задачи с использованием аппарата математики при разработке и описании алгоритма задачи.

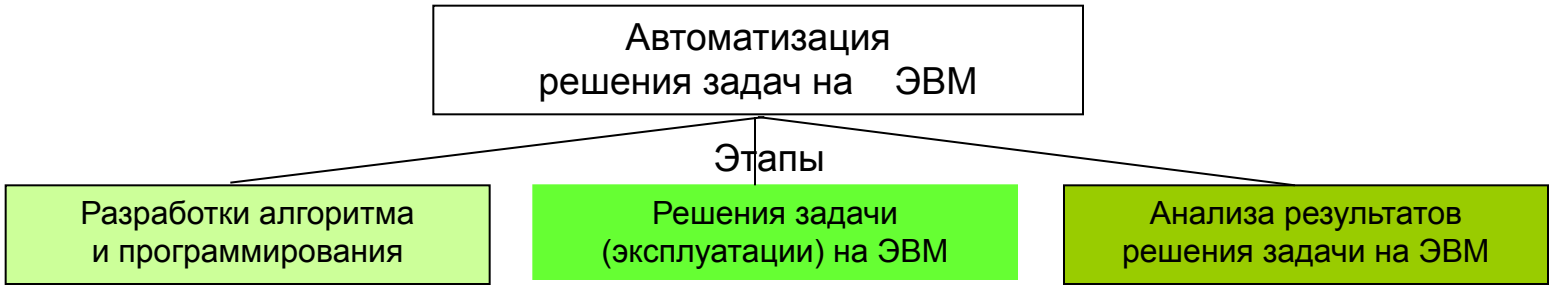
Задачи можно разделить на следующие основные типы:

- расчетные
- математические модели
- разработки текстово-графических документов
- автоматического управления
- управления базами данных
- автоматизированного принятия решений
- автоматизированного проектирования
- автоматизированного программирования
- распознавания образов
- математической лингвистики
- информационно-поисковые
- мультимедийные
- автоматизированного обучения
- информационно-справочные

С точки зрения вопросов **алгоритмизации и программирования** будут интересовать расчетные задачи и математические модели, с которыми практически все сталкиваются при использовании ЭВМ.

Расчетная задача - задача, в которой используются математические выражения для получения числовых конечных данных при ее решении.

Математическая модель - это система математических соотношений, отражающих существенные свойства объекта или явления.
Так как практически любой объект или явление бесконечны в своей сложности, то при разработке модели выделяют наиболее важные, которые отражают необходимые сущности.



2. Алгоритм, его свойства и способы представления

Алгоритм - точное описание способа решения задачи, устанавливающее состав операций и последовательность их выполнения (ГОСТ 19781-83).

Свойства алгоритма:

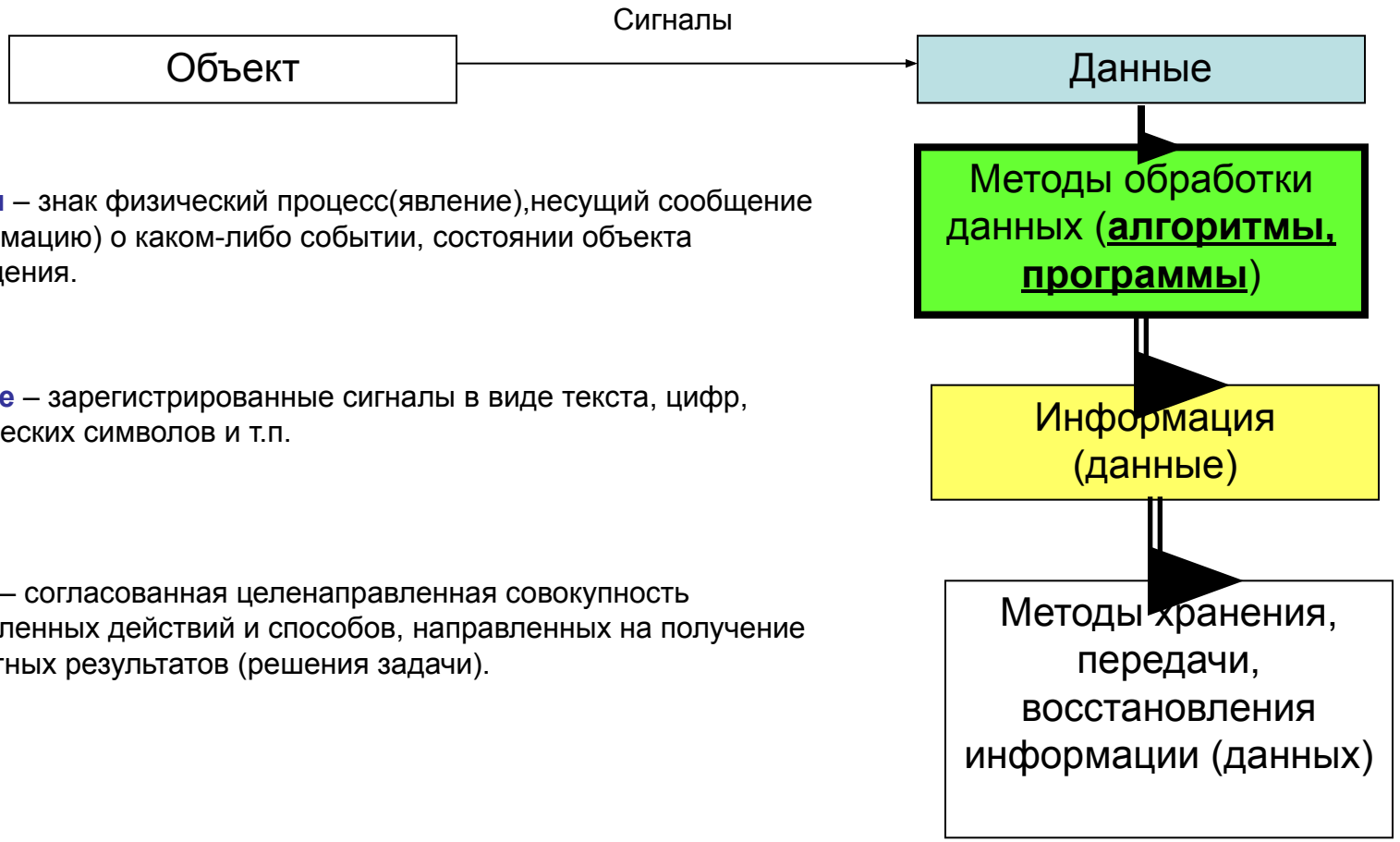
Повторяемость - получение результата при многократных расчетах с одними и теми же исходными данными.

Результативность - обязательным получением некоторого результата (числа, таблицы, текста, звука, изображения и т. д.) или сигнала о том, что данный алгоритм неприменим для решения поставленной задачи.

Массовость - возможностью получения результата при различных исходных данных для некоторого класса сходных задач.

Дискретность - возможностью разбиения алгоритма на отдельные элементарные действия.

Информацию получают в ходе информационного процесса



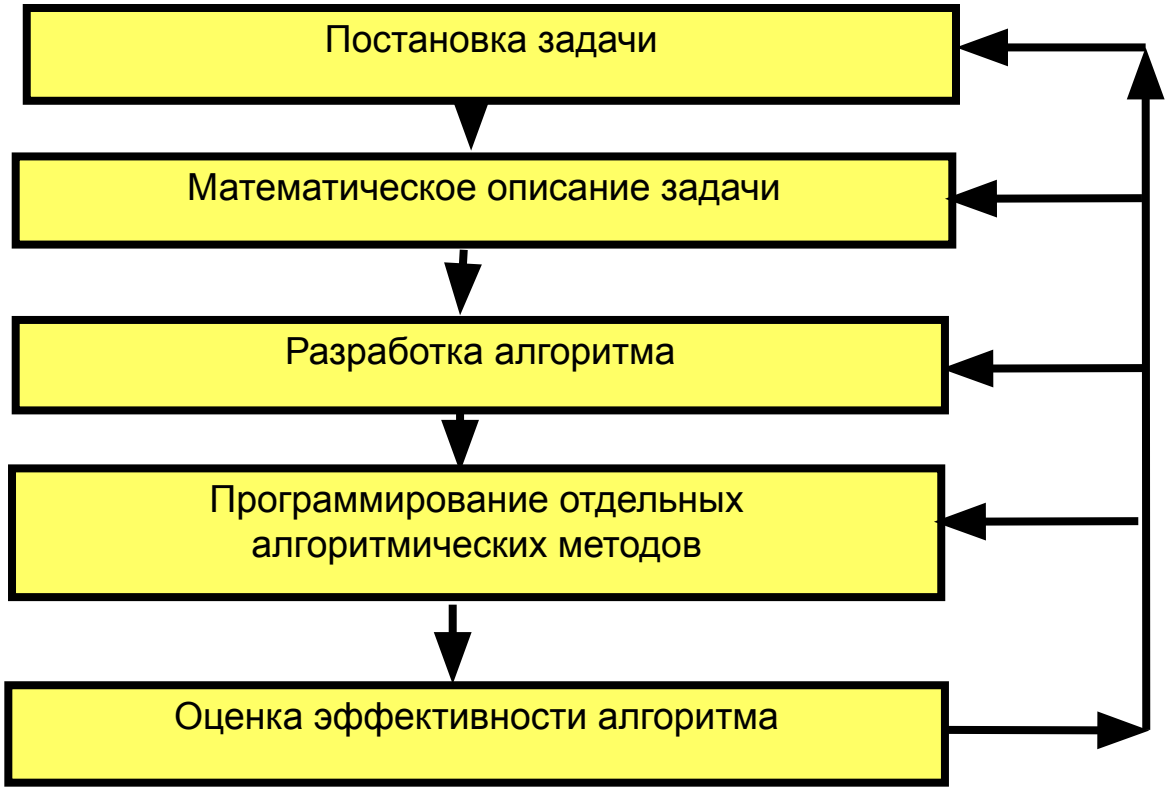
Сигнал – знак физический процесс(явление),несущий сообщение (информацию) о каком-либо событии, состоянии объекта наблюдения.

Данные – зарегистрированные сигналы в виде текста, цифр, графических символов и т.п.

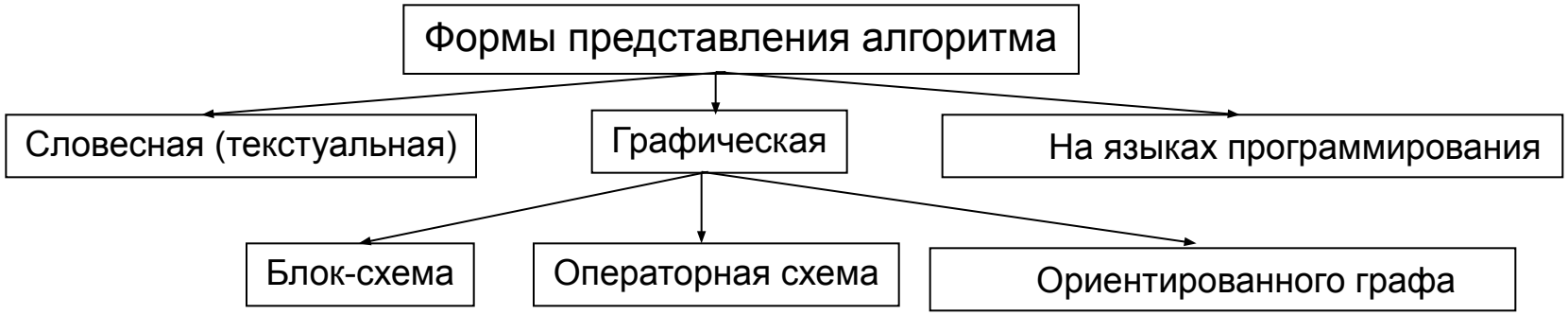
Метод – согласованная целенаправленная совокупность определенных действий и способов, направленных на получение конкретных результатов (решения задачи).

Информация – продукт взаимодействия данных и **адекватных** им методов (**алгоритмов, программ**).

Основные этапы разработки алгоритма задачи



Разработанный алгоритм решения задачи должен быть представлен в наглядной и компактной форме, удобной для практического использования и дальнейшего составления его программы.



Словесная форма применяется для представления достаточно сложных алгоритмов. Она используется лишь на начальных стадиях разработки алгоритма.

Графическая форма представления алгоритмов является более компактной и наглядной.

Блок-схема алгоритма - последовательность связанных между собой блоков, каждый из которых соответствует выполнению одного или нескольких операторов. Условные графические обозначения символов, используемых для составления блок-схемы алгоритма, стандартизированы (ГОСТ 19.002-80 и ГОСТ19.003-80).

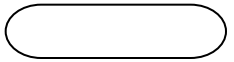
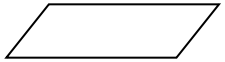
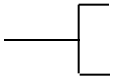
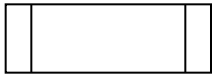
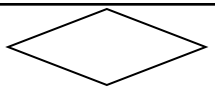
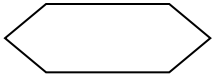
Операторная схема представления алгоритма была введена Ляпуновым. Она – цепочка символов операторов действий и условных переходов.

Ориентированный граф алгоритма - граф-схема, состоящая из конечного числа узлов (вершин) и соединяющих их ориентированных ребер. Всегда есть два узла: входной и выходной. Из каждого узла с арифметическим оператором выходит одна стрелка, а из узла с логическим оператором – две стрелки.

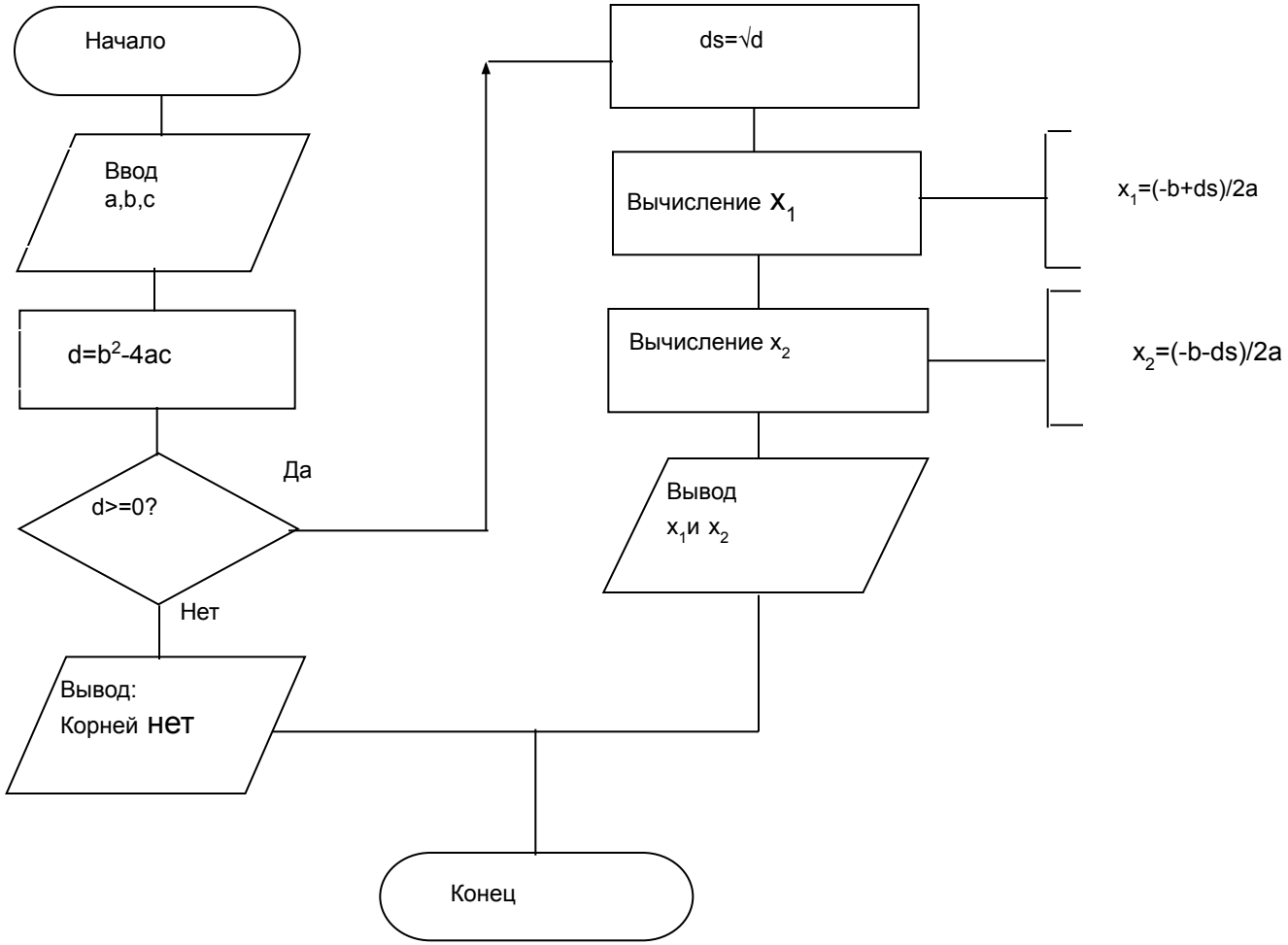
Алгоритм, представленный на языке программирования, есть программа. Алгоритм на этапе разработки может быть запрограммирован на любом удобном для его проверки языке программирования.

Блок-схема алгоритма

Основные условные графические обозначения символов, используемых для составления блок-схемы алгоритма.

Начало (Конец)	
Ввод-вывод	
Линии потока	
Комментарии	
Процесс	
Предопределенный процесс (Подпрограмма)	
Решение	
Модификация	
Внутристраничный соединитель	
Межстраничный соединитель	

Рассмотрим блок-схему алгоритма решения квадратного уравнения ax2 + bx + c = 0.

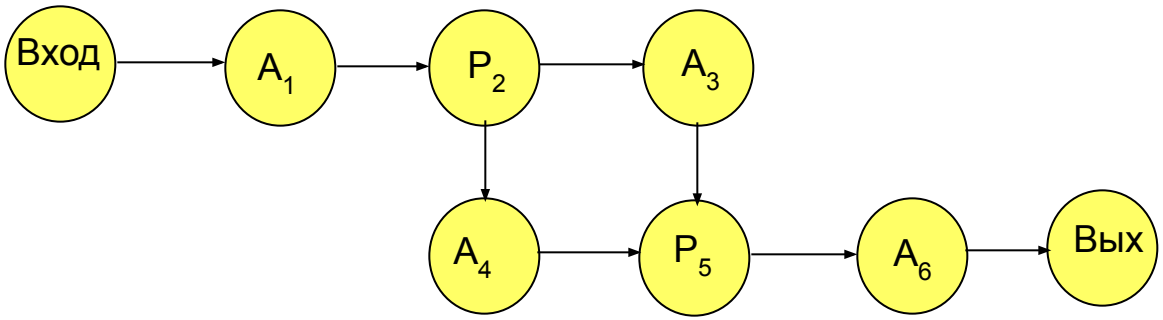


Операторная схема представления произвольного алгоритма
без раскрытия содержания операторов

Uo A1 P2 A3 ; A4 P5 A6 E7

Uo- оператор начала алгоритма
A – арифметический оператор
P – логический оператор
E – конец алгоритма

Вышеприведенный алгоритм можно представить
в виде ориентированного графа



При разработке сложных алгоритмов широко используется структурный подход

Цель структурного подхода – уменьшить сложность алгоритма за счет его декомпозиции, то есть разбиения на логические модули, выполняющие конкретные функции.

Декомпозиция алгоритма осуществляется сверху вниз (с постановки задачи).

Алгоритм в конечном итоге получается из **относительно простых логических структур**, которые можно независимо друг от друга проверить и отладить.

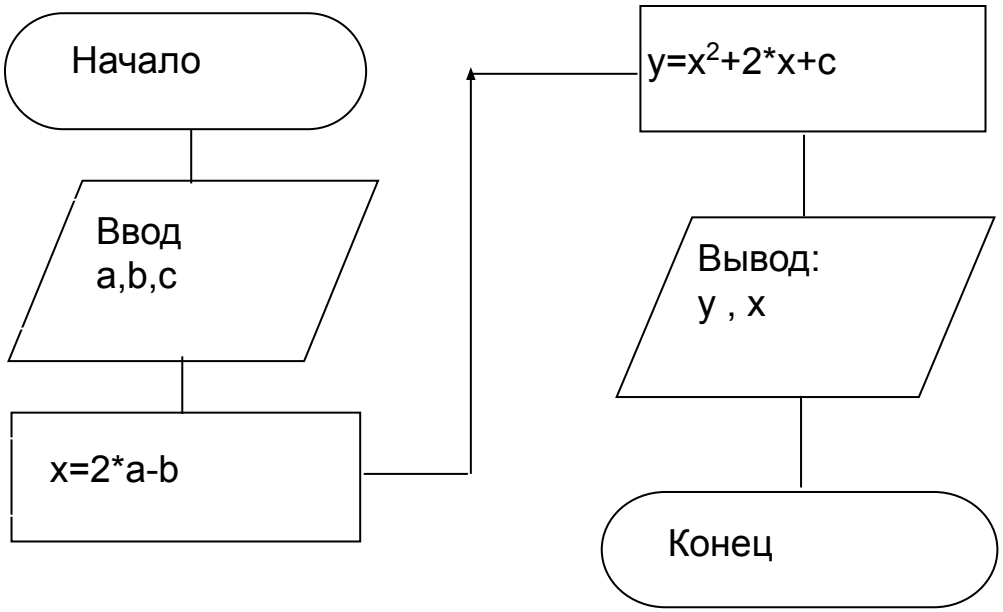
Алгоритм любой сложности может быть представлен комбинацией **трех базовых структур**:
следование;
разветвление (альтернатива, если - то - иначе);
цикл (повторение).

Характерной особенностью этих структур является наличие у них одного входа и одного выхода.

Базовая структура следование означает, что несколько операторов должны быть выполнены последовательно друг за другом и только один раз за время выполнения данной программы.

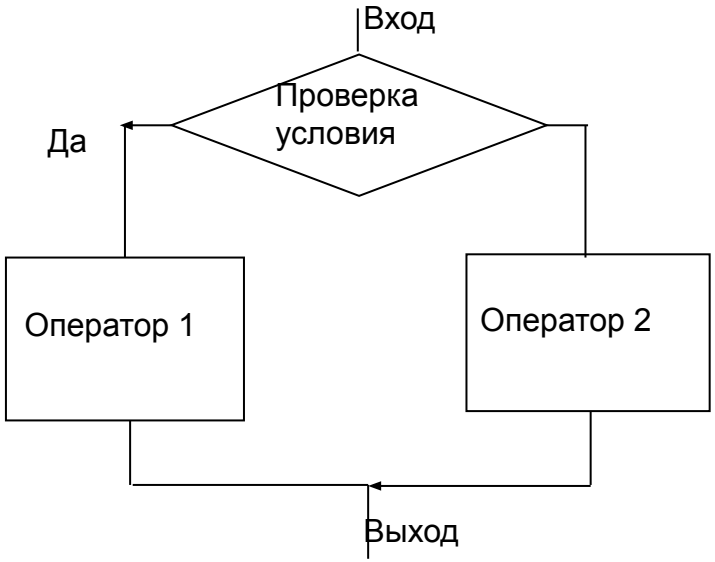
Совокупность связанных базовых структур следование называется **линейным вычислительным алгоритмом .**

Под оператором понимается формальная запись предписания для выполнения некоторой последовательности действий.



Базовая структура
следование

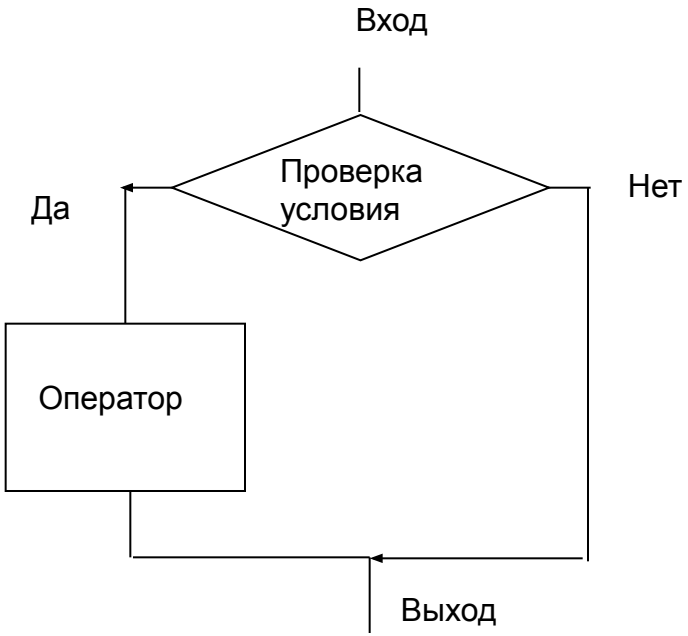
Базовая структура разветвление (если - то - иначе) обеспечивает, в зависимости от результата проверки условия (истина или ложь), выбор одного из альтернативных путей работы алгоритма, причем каждый из путей ведет к общему выходу.



Базовая структура разветвление

Алгоритм, в состав которого входит базовая структура разветвление, называется **разветвляющимся**.

В частном случае может оказаться, что для одного из выбранных путей действий предпринимать не нужно. Такая структура получила название **обход** или структура «**если – то».**

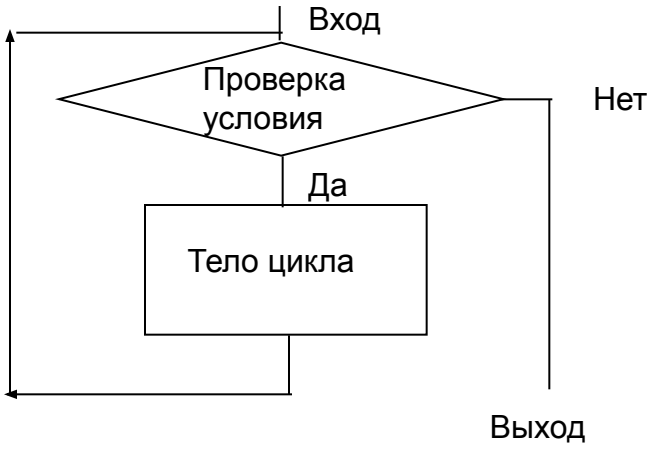


Базовая структура обход

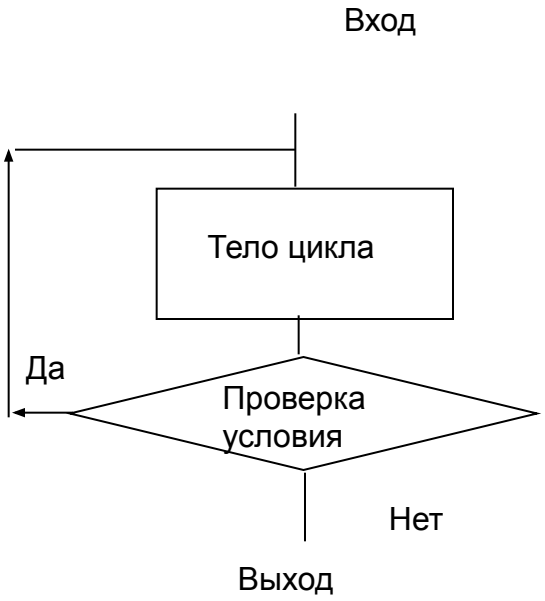
Если в алгоритме имеется три и более направления ветвления, то его можно представить в виде совокупности нескольких базовых структур разветвление «если - то – иначе».
Такую разновидность структуры разветвление часто называют **множественный выбор**.

Базовая структура цикл обеспечивает повторное выполнение или, другими словами, циклическую работу операторов.

Различают две разновидности этой структуры: «**цикл – пока»** и «**цикл – до»**.



Базовая структура «цикл-пока»



Базовая структура «цикл-до»

Алгоритмы, имеющие в своем составе базовую структуру цикл, называются **циклическими**.

3. Разработка программ

В соответствии с действующей для данной ЭВМ системой программного обеспечения алгоритм решения задачи записывается либо на **языке вычислительной машины**, либо на **алгоритмическом языке**.

Программирование алгоритмов осуществляется **вручную** и с помощью **автоматизированных систем программирования.**

Программа - упорядоченная последовательность команд.

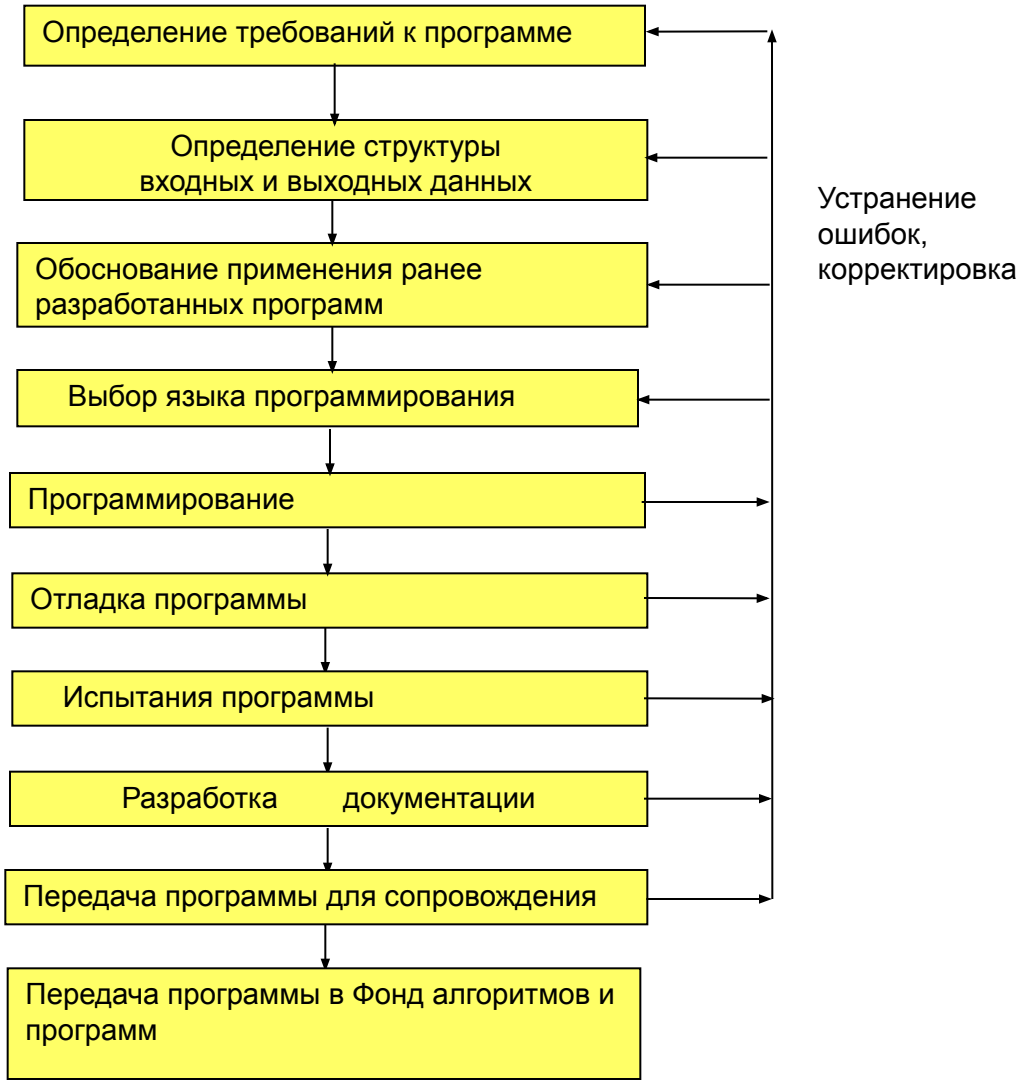
ГОСТ 19.1001-77 устанавливает следующие **виды программ**:
компонент;
комплекс.

Компонент- программа, рассматриваемая как единое целое, выполняющая законченную функцию и применяемая самостоятельно или в составе комплекса.

Комплекс - программа, состоящая из двух или более компонент и (или) комплексов, выполняющих взаимосвязанные функции, и применяемые самостоятельно или в составе другого комплекса.

Приложение (операционной системы) - программа, функционирующая под управлением конкретной операционной системы.

Этапы разработки программ определяются ГОСТ 19.102-77.



Этапы разработки программ

Характерные ошибки при разработке программ

Вид ошибки	Характеристика ошибки
Неверная постановка задачи	Неверно сформулирована задача
Ошибка в анализе исходных данных и начальных условиях задачи	Неполный учет ситуаций, влияющих на ход решения задачи
Неверный алгоритм	Выбор методов, не обеспечивающих решение задачи по заданным условиям
Синтаксическая ошибка	Нарушение правил, определяемых языком программирования
Семантическая ошибка	Неверная трактовка порядка выполнения операторов
Ошибка при выполнении вычислений	Деление на ноль, извлечение корня из отрицательного числа, слишком большое или маленькое число и т.п.
Ошибка в данных	Неправильно определен диапазон данных
Ошибки ввода-вывода	Неверное считывание данных, неправильное задание форматов и т.п.
Опечатки	Перепутаны похожие символы

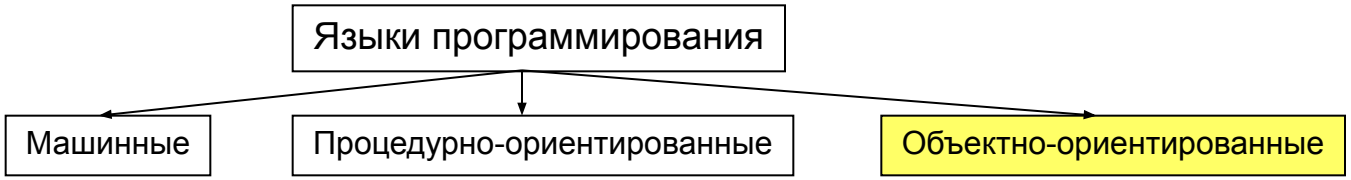
Для создания эффективной программы нужно четко знать назначение и особенности языков программирования.

Языки программирования являются искусственными языками, в них синтаксис и семантика строго определены.

Языки программирования, в отличие от естественных языков, не допускают многозначных и произвольных толкований.

Синтаксис - это набор правил, которые определяют основные внутренние структуры и последовательности символов, допустимых в языке программирования.

Семантика - это значения языковых единиц (слов, словосочетаний, предложений).



Машинный язык - свод правил кодирования действий ЭВМ с помощью двоичных чисел.
Для упрощения записи программ или представления данных двоичная система часто заменяется восьмеричной или шестнадцатеричной.

Процедурно-ориентированные языки используют буквенные обозначения (символы) для замены машинных кодов ЭВМ, а также имеют возможности для автоматизации процесса распределения памяти, диагностики ошибок и трансляции программы.

Транслятор осуществляет перевод программы с процедурно-ориентированного языка на язык команд процессора ЭВМ.

В основе **объектно-ориентированных языков** лежит понятие **объект**.

Объекты представляют собой многократно используемые программные модули.

Структурно объекты состоят из двух частей: **свойств** и **методов**.

Свойства являются **параметрами объекта**.

Методы определяют **поведение объекта**, его возможности.

Все однотипные объекты объединяются в **классы**, где первоначально описываются свойства и разрабатываются методы.

Класс является типом объекта как, например, для чисел вводятся целый или вещественный тип. Причем тип данных определяет множество допустимых значений этих данных и множество разрешенных операций над ними. Также и класс определяет возможности объекта.

Объектно-ориентированное программирование характеризуется следующими тремя признаками: **инкапсуляцией, наследованием и полиморфизмом.**

Инкапсуляция – объединение свойств и методов в одном объекте. Целью инкапсуляции является обеспечение автономности объектов и возможности изменения их структуры без внесения изменений в другие части программы.

При инкапсуляции объект заключается в непроницаемую оболочку, и только его внешний вид доступен для обозрения. Этим обеспечивается защита объекта от кодов других частей программы. Объект отвечает за корректность реализации своей функциональной способности, а вызывающая объект программа - за корректность использования объекта.

С помощью механизма наследования одни классы объектов могут происходить от других. Дочерний класс способен унаследовать от своего родительского класса все его методы и данные, причем потомок может унаследовать способности и от нескольких родителей.

То есть наследование позволяет использовать разработанные ранее классы, что обеспечивает сокращение разработки программ. При этом расширяются возможности класса за счет объединения свойств и методов исходных классов.

Полиморфизм – позволяет использовать одно имя для обозначения действий, общих для родственных классов. При этом конкретизация выполняемых действий осуществляется в зависимости от типа обрабатываемых данных. Полиморфизм исключает избыточность кодов в программе.

Этапы подготовки и выполнения программы в интегрированной среде программирования

Программа с языка высокого уровня с помощью текстового процессора, переводиться **компилятором** на машинный язык.

Интегрированная среда программирования представляет собой программу, имеющую редактор текстов, компилятор, отладчик, компоновщик, подсистему работы с файлами, справочную систему.

Компоновщик в программу вставляются коды стандартных команд и библиотечных функций