

Курс «Основы программирования»

Федорук Елена Владимировна ст. преподаватель каф РК-6 МГТУ
им.Н.Э.Баумана

Лекция №2

Алгоритм

Алгоритм — формально описанная последовательность действий, которые необходимо выполнить для получения требуемого результата.

Основные особенности алгоритма

- конечность (алгоритм всегда должен заканчиваться после выполнения конечного числа шагов);
- определенность (каждый шаг алгоритма должен быть точно определен);
- ввод (алгоритм имеет некоторое, возможно равное нулю, число входных данных, т.е. величин, которые задаются до начала его работы или определяются динамически во время его работы);
- вывод (у алгоритма есть одно или несколько выходных данных, т.е. величин, которые имеют вполне определенную связь с входными данными);
- эффективность (алгоритм обычно считается эффективным, если все его операторы достаточно просты для того, чтобы их можно было выполнить в течение конечного промежутка времени с помощью карандаша и бумаги).

Одним из критериев качества алгоритма является время, необходимое для его выполнения. Данную характеристику можно оценить по тому, сколько раз выполняется каждый шаг. Другими критериями является адаптируемость алгоритма к различным компьютерам, его простота, изящество и т.д.

Различают последовательности действий линейной, разветвленной и циклической структуры.

Линейная структура представляет из себя определенную последовательность операций, выполняемых друг за другом.

Для **разветвленной структуры** конкретная последовательность операций зависит от значений одного или нескольких параметров.

Циклическая структура включает в себя многократно повторяющуюся последовательность операций.

Циклические структуры можно разделить следующим образом:

- **счетные циклы** — циклические процессы, для которых количество повторений известно;
- **итерационные циклы** — циклические процессы, завершающиеся по достижении или нарушении некоторых условий;
- **поисковые циклы** — циклические процессы, из которых возможны два варианта выхода: выход по завершению процесса и досрочный выход по какому-либо дополнительному условию.

Формальное описание алгоритмов осуществляется с использованием схем алгоритмов и псевдокодов.

Схемы алгоритмов

Схемы используются для формального описания алгоритмов. На изображение *схем алгоритмов* существует ГОСТ 19.701-90, согласно которому каждой группе действий соответствует блок особой формы.

Наиболее часто используемые элементы схем алгоритмов:


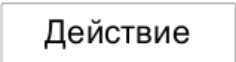


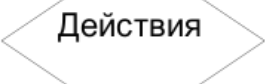


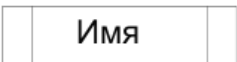

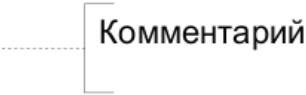
Название блока	Обозначение	Назначение блока
Терминатор		Начало, завершение программы или подпрограммы
Процесс		Обработка данных
Данные		Операции ввода-вывода
Решение		Ветвления, выбор, итерационные и поисковые циклы
Подготовка		Счетные циклы
Граница цикла		Любые циклы
		
Предопределенный процесс		Вызов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

Рис. 1. Основные элементы схем алгоритмов

При разработке алгоритма каждое действие обозначается соответствующим блоком, показывая их последовательность линиями со стрелками на конце. Желательно, чтобы линия входила сверху, а выходила снизу. Если линии идут не слева направо и не сверху вниз, то стрелка в конце линии обязательна. В противном случае ее можно не ставить.

В случае, когда схема алгоритма не уместится на листе, используют соединители. При переходе на другой лист или получении управления с другого листа в комментариях указывается номер листа, например, "с листа 2", "на лист 1".

Основные алгоритмические структуры

В теории программирования доказано, что для записи любого, сколь угодно сложного алгоритма достаточно трех базовых структур:

- **следование** (последовательное выполнение действий);
- **ветвление** (выбор одного из двух вариантов действий);
- **цикл-пока** (повторение действий, пока не будет нарушено условие, выполнение которого проверяется в начале цикла).

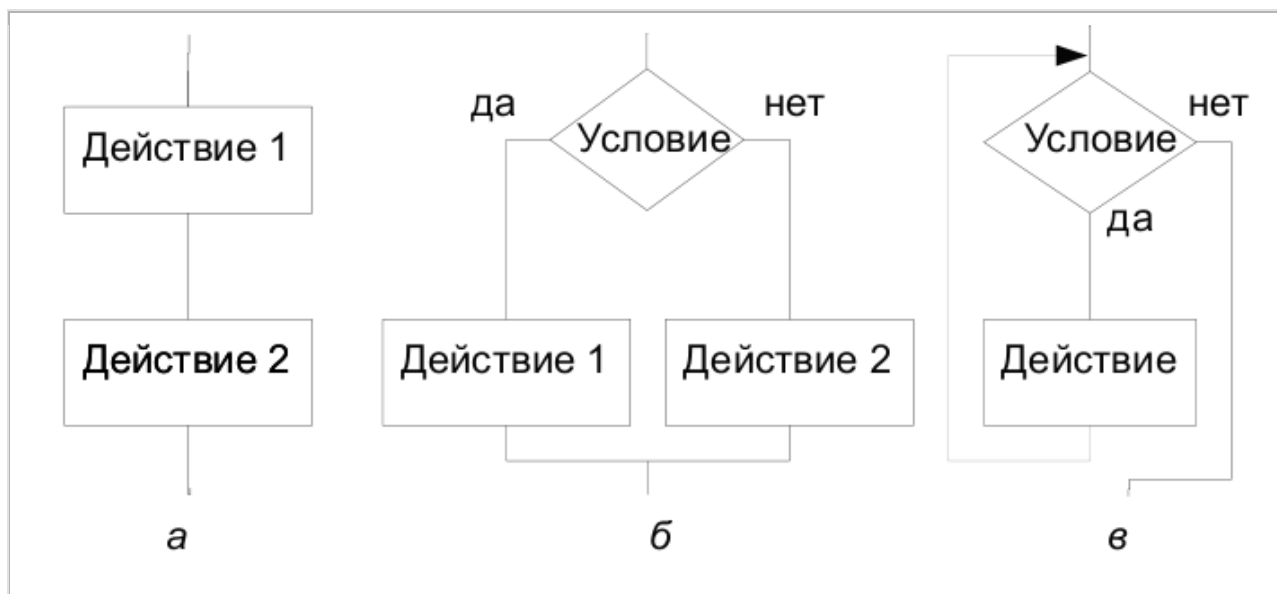


Рис. 1. Базовые алгоритмические структуры: следование (а), ветвление (б), цикл-пока (в)

Помимо базовых структур существуют дополнительные структуры, производные от базовых:

- **выбор** (выбор одного варианта из нескольких в зависимости от значения некоторой величины);
- **цикл-до** (повторение действий, пока не будет нарушено условие, выполнение которого проверяется в конце цикла);
- **счетный цикл** (цикл с заданным числом повторений).

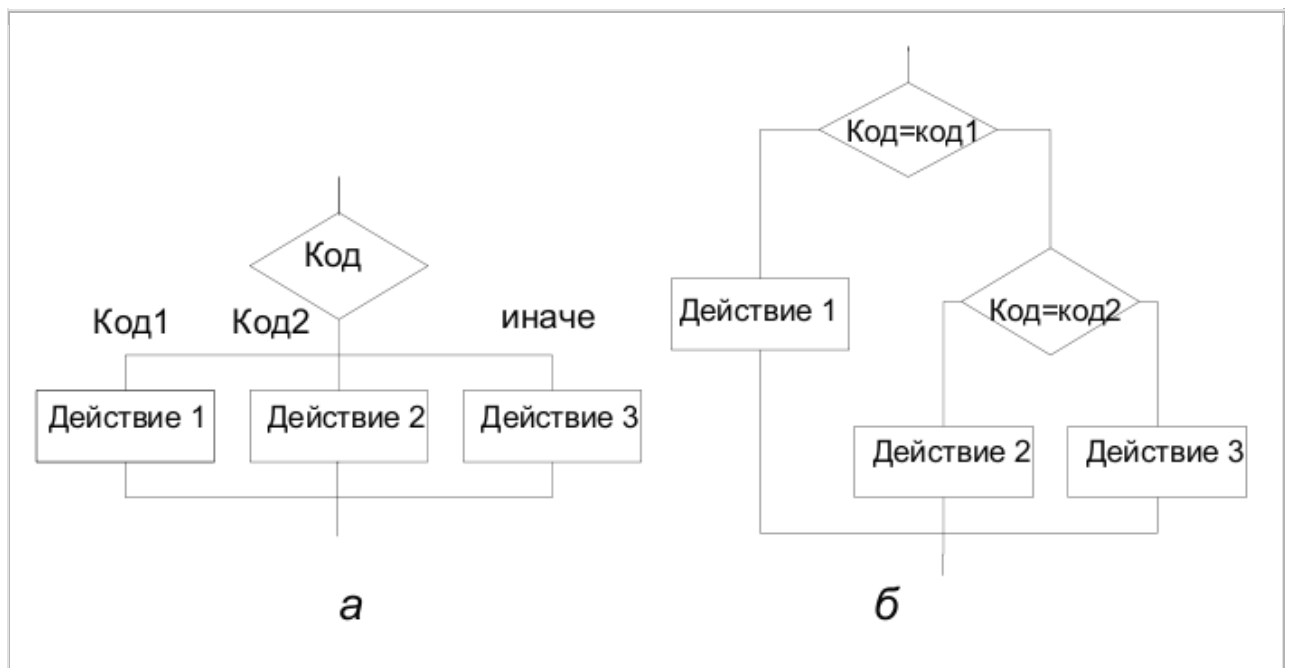


Рис. 2. Дополнительная структура выбор (а) и ее реализация через базовые структуры (б)

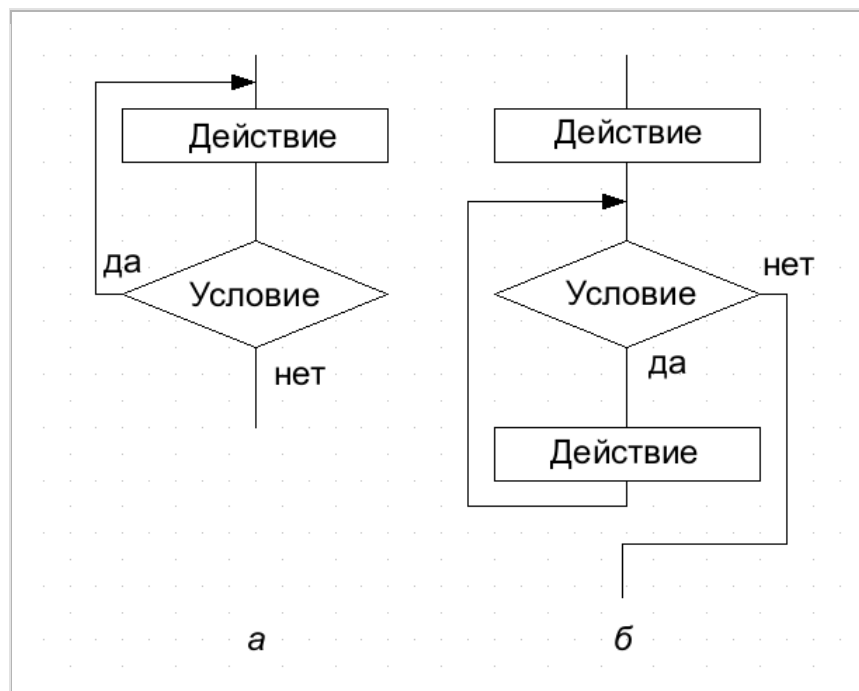


Рис. 3. Дополнительная структура цикл-до (а) и ее реализация через базовые структуры (б)

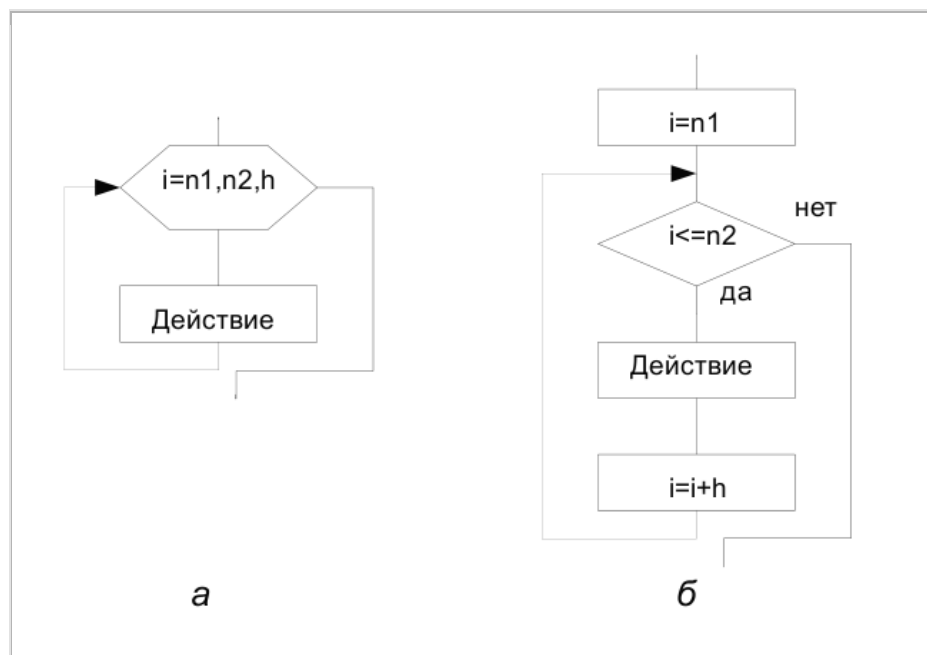


Рис. 4. Дополнительная структура счетный цикл (а) и ее реализация через базовые структуры (б)

Перечисленные структуры были положены в основу структурного программирования. В том случае, если в схеме алгоритма отсутствуют другие варианты передачи управления, алгоритм называется **структурным**.

Псевдокод алгоритма

Псевдокод алгоритма — один из способов формального описания алгоритма. Псевдокод базируется на тех же основных структурах, что и структурные алгоритмы. Описать на псевдокоде неструктурный алгоритм нельзя.

Для каждой структуры алгоритма используют свою форму описания. Известно несколько вариантов форм псевдокодов. Один из вариантов приведен ниже:

Следование

<действие1>

<действие2>

Выбор

Выбор <код>

 <код1>:<действие1>

 <код2>:<действие2>

Ветвление

Если <условие>

 то <действие1>

 иначе <действие2>

Все-если

Цикл-пока

Цикл-пока <условие>

 <действие>

Все-цикл

Счетный цикл

Для <индекс>=<n>, <k>, <h>

 <действие>

Все-цикл

Цикл-до

Выполнять

 <действие>

До <условие>

Пример алгоритма

Разработать алгоритм для вычисления действительных и комплексных корней квадратного уравнения

$$ax^2+bx+c=0$$

Коэффициенты a, b, c вводятся в начале работы программы.

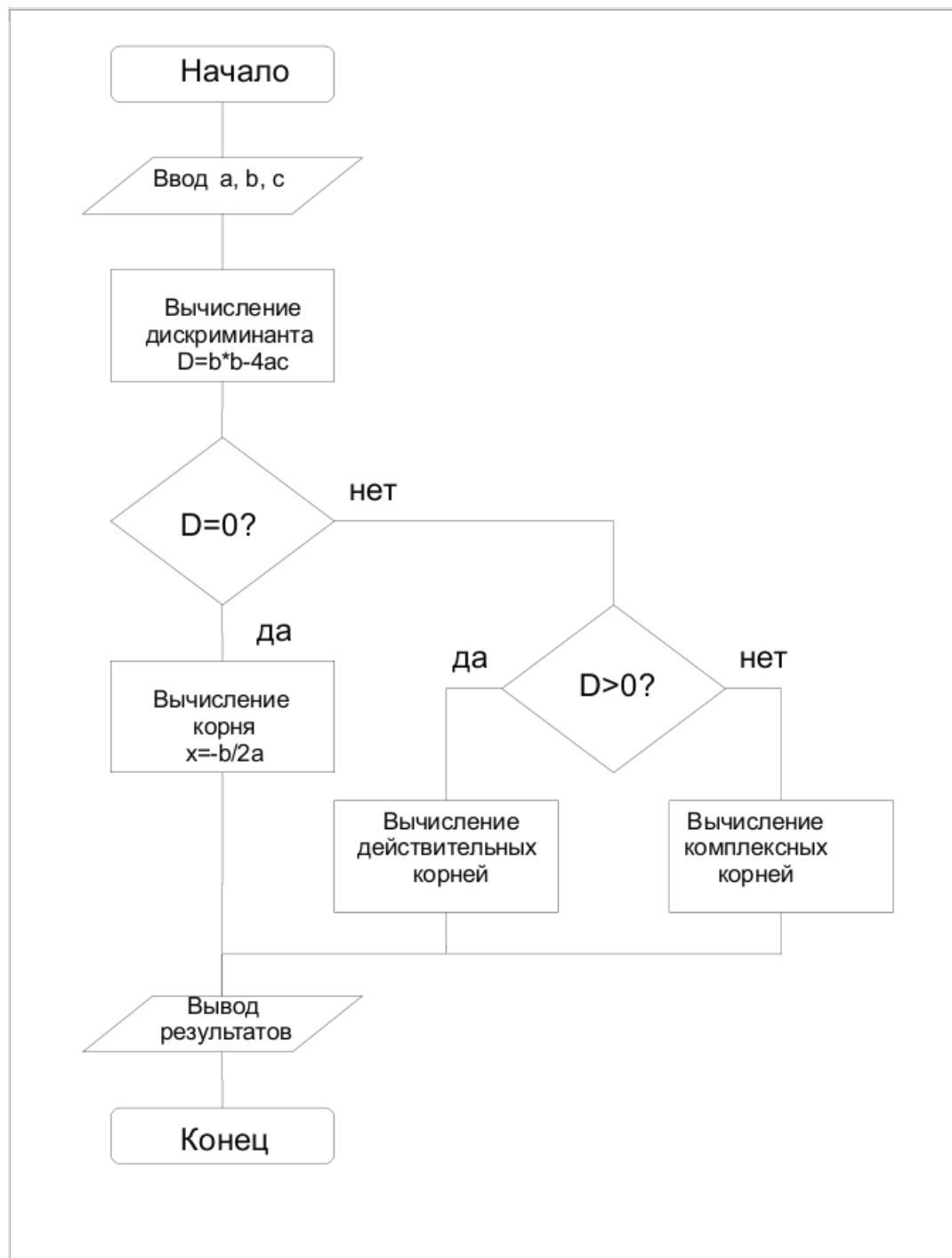


Рис. 1. Схема алгоритма

```
Алгоритм вычисления корней квадратного уравнения
Ввести коэффициенты квадратного уравнения a, b, c.
Вычислить дискриминант
 $D=b^2-4ac$ 
Если  $D=0$ 
    Вычислить единственный действительный корень
     $x=-b/2a$ 
Иначе
    Если  $D>0$ 
        Вычислить значения действительных корней
         $x1=(-b+\sqrt{D})/2a$ 
         $x2=(-b-\sqrt{D})/2a$ 
    Иначе
        Вычислить значения комплексных корней
         $x1=-b/2a + i*\sqrt{-D}/2a$ 
         $x2=-b/2a - i*\sqrt{-D}/2a$ 
Все-если
Все-если
Вывести результаты вычислений
Конец алгоритма
```

Рис. 2. Псевдокод алгоритма