

# **Алгоритмизация и ОСНОВЫ программирования**

# Алгоритмы для ЭВМ

# **1. Основные понятия**

# АЛГОРИТМ –

это точное предписание о  
последовательности  
действий, которые должны  
быть произведены для  
получения результата

# Алгоритмический язык -

это формальный язык для записи алгоритмов, который включает в себя набор символов (алфавит языка), систему правил связи символов для образования «слов», с помощью которых представляются отдельные составляющие компоненты языка (синтаксис языка), и систему правил истолкования слов языка (семантику).

# 2. Свойства алгоритмов

# 1. Дискретность алгоритма

Свойство алгоритма, означающее, что процесс решения задачи, определяемый алгоритмом, расчленён на отдельные элементарные действия (шаги) и, соответственно, алгоритм представляет последовательность указаний, команд, определяющих порядок выполнения шагов процесса.

## 2. Определённость алгоритма

Это свойство означает, что каждая команда алгоритма должна быть понятна исполнителю, не оставлять места для её неоднозначного толкования и неопределённого исполнения.



# 3. Результативность алгоритма

Свойство алгоритма, состоящее в том, что он всегда приводит к результату через конечное, возможно, очень большое число шагов.

## **4. Массовость алгоритма**

**каждый алгоритм,  
разработанный для решения  
некоторой задачи, должен  
быть применим для решения  
задач этого типа при всех  
допустимых значениях  
исходных данных.**

## **3. Элементы для задания алгоритма:**

- **набор объектов, составляющих совокупность возможных исходных данных, промежуточных и конечных результатов;**
- **правило начала;**
- **правило непосредственной переработки информации (описание последовательности действий);**
- **правило окончания;**
- **правило извлечения результатов.**

# 4. Виды алгоритмов:

- **Линейный алгоритм** – описание действий, которые выполняются однократно, при этом четко друг за другом;
- **Разветвляющийся алгоритм** – алгоритм, в котором в зависимости от условия выполняется либо одна, либо другая последовательность действий.
- **Циклический алгоритм** – описание действий, которые должны повторятся определенное количество раз или пока не выполнится условие.

## Разветвляющийся алгоритм может быть полной и неполной формы

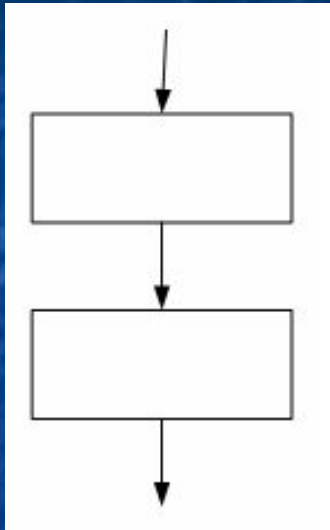
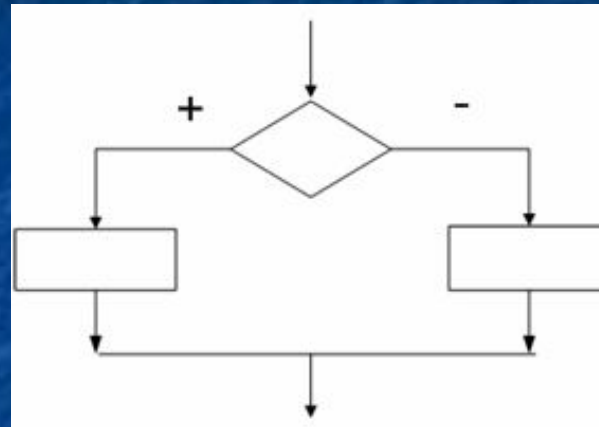
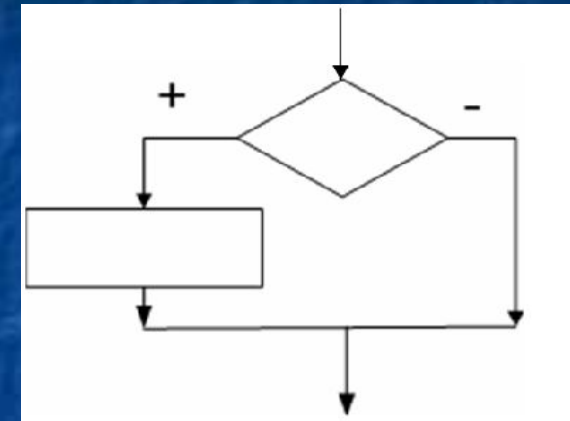


Схема  
линейного  
алгоритма



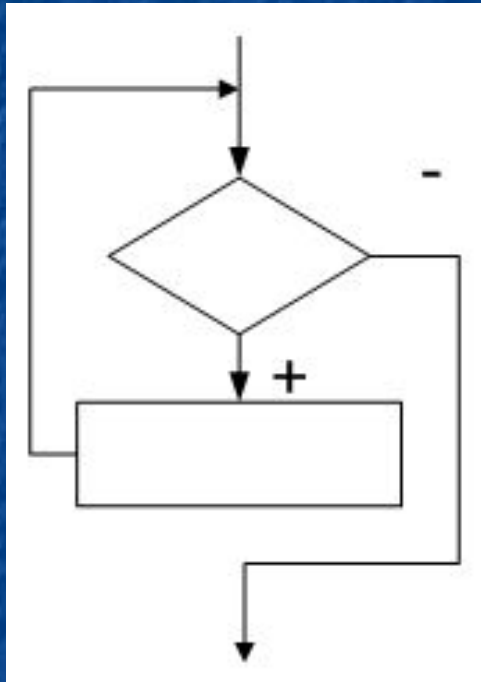
полная форма



неполная форма

Неполная форма команды ветвления используется тогда, когда необходимо выполнять действие только в случае соблюдения условия. Если условие не соблюдается, то команда ветвления завершает свою работу без выполнения действия.

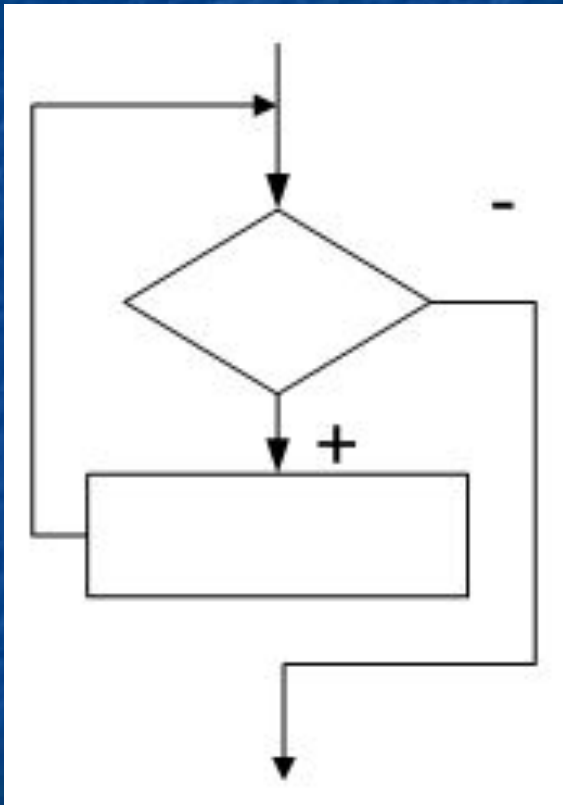
## а) цикл с предусловием



В цикле с предусловием, называемом циклом "пока", сначала проверяется условие, а затем выполняется действие. Цикл "пока" работает так: пока условие выполняется, выполняется тело цикла. Характерно, что тело цикла в цикле "пока" может не выполниться ни разу (если условие сразу не выполнится).

# Циклический алгоритм

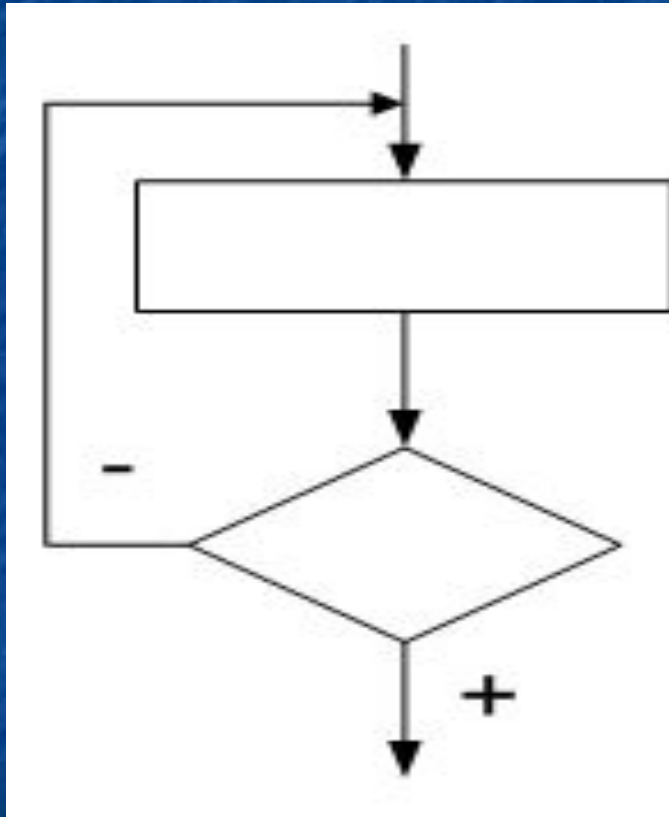
## а) цикл с предусловием



В цикле с предусловием, называемом циклом "пока", сначала проверяется условие, а затем выполняется действие. Цикл "пока" работает так: пока условие выполняется, выполняется тело цикла. Характерно, что тело цикла в цикле "пока" может не выполниться ни разу (если условие сразу не выполнится).

# Циклический алгоритм

## б) цикл с постусловием



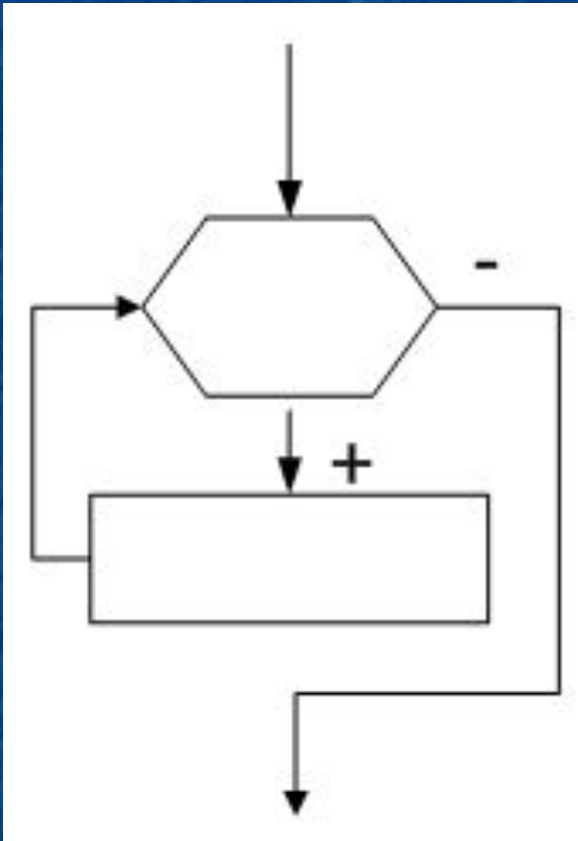
В цикле с постусловием, называемом циклом "до", наоборот: сначала выполняется действие, а лишь потом проверяется условие.

Цикл "до" функционирует следующим образом: до тех пор, пока условие не выполнится, выполняется тело цикла. Характерно, что тело цикла в цикле "до" выполняется хотя бы один раз.



# Циклический алгоритм

## в) цикл с параметром



Структура данного цикла иначе называют циклом  $i$  раз. Эта команда выполняется таким образом: параметру  $i$  присваивается начальное значение  $a$ , сравнивается с конечным значением  $b$  и, если оно меньше или равно конечному значению  $b$ , выполняется серия команд. Параметру присваивается значение предыдущего, увеличенного на величину  $h$  – шага изменения параметра и вновь сравнивается с конечным значением  $b$

# Основы программирования

# **1. Основные понятия**

# Программирование -

**это наука, изучающая теорию и методы разработки, производства и эксплуатации программного обеспечения ЭВМ.**

# **Язык программирования -**

**это способ записи программ  
решения различных задач  
на ЭВМ в понятной для  
компьютера форме.**




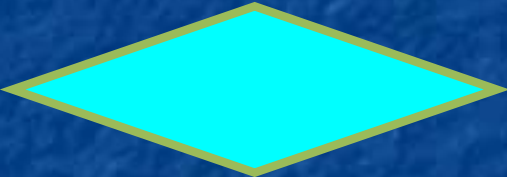


# Языки программирования:

**БЕЙСИК, ФОРТРАН,  
КОБОЛ, ПАСКАЛЬ, СИ,  
СИ+, СИ++ и др.**

## 2. Этапы подготовки и решения задач на ЭВМ

1. Постановка задачи.
2. Математическое описание задачи.
3. Выбор и обоснование метода решения.
4. Алгоритмизация вычислительного процесса.
5. Составление программы.
6. Отладка программы.
7. Решение задачи на ЭВМ и анализ результатов.

# 3. Обозначение элементов блок-схем

 <p><b>Начало</b></p>	Начало алгоритма
 <p><b>Конец</b></p>	Конец алгоритма
	Выполняемое действие
	Ветвление программы
	Счетчик количества повторов
	Последовательность выполнения действий



## 4. Задача:

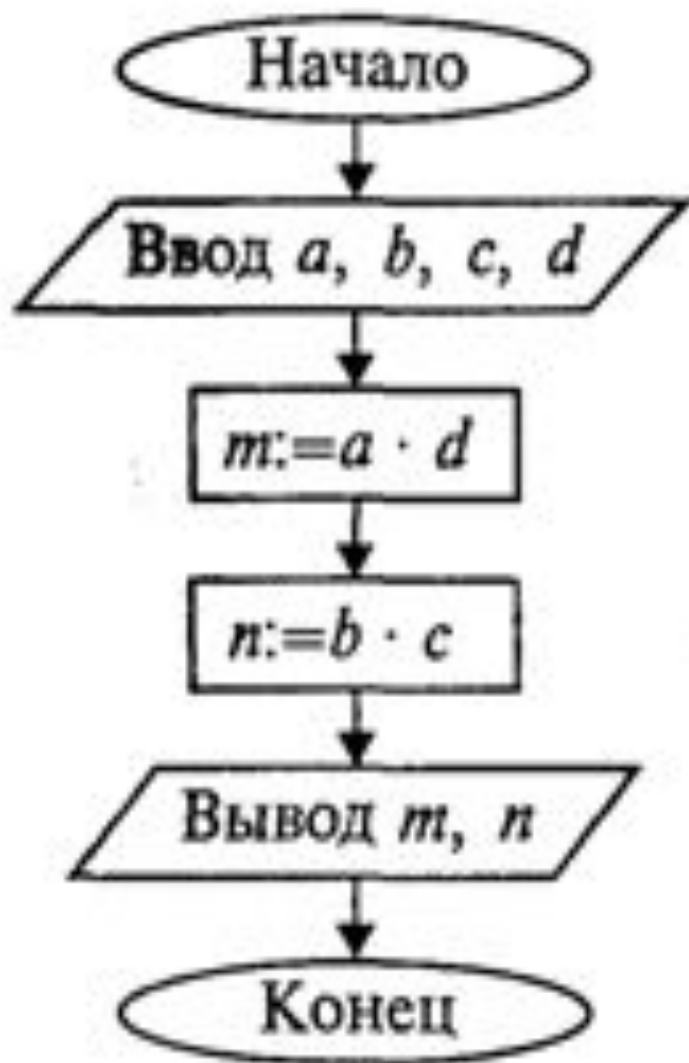
Правила деления обыкновенных дробей описаны так:

- 1.** Числитель первой дроби умножить на знаменатель второй дроби.
- 2.** Знаменатель первой дроби умножить на числитель второй дроби.
- 3.** Записать дробь, числитель которой есть результат выполнения пункта 1, а знаменатель — результат выполнения пункта 2. В алгебраической форме это выглядит:

$$\frac{a}{b} : \frac{c}{d} = \frac{a \cdot d}{b \cdot c} = \frac{m}{n}.$$

Построим алгоритм деления дробей для ЭВМ. В этом алгоритме сохраним те же обозначения для переменных, которые использованы в записанной выше формуле. *Исходными данными* являются целочисленные переменные *a, b, c, d*. *Результатом* — также целые величины.

# Блок-схема и текст алгоритма на алгоритмическом языке (АЯ):



```
алг Деление дробей
нач
    цел a, b, c, d, m, n
    ввод a, b, c, d
    m := a * d
    n := b * c
    вывод m, n
кон
```

**Паскаль. Циклы.  
Решение задач с  
ПОМОЩЬЮ ЦИКЛОВ.**

# ЦИКЛЫ

В Паскале три вида циклов:

- Цикл **For** (со счетчиком);
- Цикл **While** (с предусловием);
- Цикл **Repeat** (с постусловием).

# Цикл FOR

*For* позволяет выполнить серию действий заданное число раз.

Общий формат :

*for i:=nz to kz do шаг 1*

**тело цикла**

ИЛИ

*for i:=nz downto kz do шаг -1*

**тело цикла**

где **i-переменная цикла или счетчик** и должна быть только целого типа, **nz- начальное значение** переменной, **kz – конечное значение** переменной программы.

# Цикл While

Общий формат:

*while условие do*

*begin*

*тело цикла*

*end;*

где *условие* - некоторое выражение, результат может "истина" или "ложь".

*Если в теле цикла несколько операторов, то их нужно объединить программными скобками begin – end.*

# Как работает *While*? Сначала проверяется условие

Если оно истинно, то тело цикла выполняется, затем условие проверяется снова, и процесс повторяется. Тело цикла выполняется каждый раз, когда проверка условия дает "истину".

Если условие ложно, то цикл завершается, входа в тело цикла не происходит, и следующим выполняется предложение, стоящее непосредственно после цикла.

После *while* ставится условие *работы* цикла. Проверка условия производится до выполнения цикла.



# Цикл Repeat

Общий  
формат:

**repeat**

**тело цикла**

**until**

**условие**

Сначала выполняется тело цикла.

По достижению пункта *until* проверяется условие.

Если оно не выполняется, тело цикла выполняется снова, с новой проверкой условия.

Если же условие выполнено, то тело цикла больше не повторяется, цикл завершается, а программа переходит к выполнению оператора, следующего за *until*.

В этом цикле не используются программные скобки *begin-end*.

После *until* ставится условие *выхода* из цикла.

# Задача

Даны равносторонние треугольники. Значение сторон меняются от 10 до 24 с шагом 1. Вывести периметры всех треугольников.

# Цикл FOR

```
for i:=10 to 24 do  
  Begin  
    P:=i*3;  
    writeln('i= ',i,'P= ',P);  
  end;  
end.
```

# Цикл WHILE

```
i:=10;  
while i<=24 do  
begin  
  P:=i*3;  
  writeln('i= ',i,'P= ',P);  
  i:=i+1;  
end;  
end.
```

# Цикл Repeat

```
var i,P: integer;  
begin  
  i:=10;  
  repeat  
    P:=i*3;  
    writeln('i= ',i,' P= ',P);  
    i:=i+1;  
  until i=25;  
end.
```