

# Разветвляющиеся алгоритмы и программы



**ЕСЛИ** пошел дождь,  
**ТО** надо открыть зонт

**ВСЕ**

**ЕСЛИ** назвался груздем,  
**ТО** полезай в кузов

**ВСЕ**

**ЕСЛИ** ласточки летают низко,  
**ТО** будет дождь,  
**ИНАЧЕ** дождя не будет

**ВСЕ**

**Разветвляющимся** называется алгоритм (программа), в котором в зависимости от истинности или ложности некоторого условия, выбирается один из двух (или нескольких) возможных путей продолжения алгоритма.

Эти пути продолжения называют **ветвями**

В зависимости от количества ветвей различают:

1. **Полное ветвление** (две ветви)
2. **Неполное ветвление** (одна ветвь)
3. **Многократное ветвление** (больше двух ветвей)

## Полное ветвление



В качестве **условия** указывается некоторое логическое выражение. Если условие оказывается **истинным**, то выполняется **действие1**, в **противном** случае выполняется **действие2**

## Неполное ветвление



Если **условие** оказывается **истинным**, то выполняется **действие**, в противном случае происходит переход к следующему оператору программы.

В языке Паскаль для реализации разветвляющихся программ используется *условный оператор*.

Он имеет следующий *формат*:

```
If <логическое выражение>  
    then <оператор 1>  
    [else <оператор2>];
```

Если в условном операторе присутствуют обе ветви (**then** и **else**), то такую форму оператора называют *полной*, если же ветвь **else** отсутствует – *неполной*.



При выполнении условного оператора в полной форме вычисляется значение логического (булевского) выражения. Если оно равно 'true', то выполняется <оператор 1>, а <оператор 2> пропускается; если же оно равно 'false', то выполняется <оператор 2>, а <оператор 1> пропускается.

При выполнении условного оператора в неполной форме также сначала вычисляется значение булевского выражения. Если оно равно 'true', то выполняется <оператор 1>, если 'false' - осуществляется переход к оператору, следующему за условным оператором, то есть в последнем случае <оператор 1> просто пропускается (“обходится”).

После ключевых слов *then u else* по правилам языка Паскаль можно записывать только один оператор, а часто бывает необходимо выполнить несколько операторов на каждой из ветвей. Тогда эти операторы объединяются в один **составной оператор** при помощи **операторных скобок**, роль которых выполняют ключевые слова *begin* (открывающаяся скобка) и *end* (закрывающаяся скобка). То есть *составной оператор* имеет следующий *формат*:

**begin**

**<оператор 1>;**

**<оператор 2>;**

**.....**

**<оператор n>**

**end;**

Если в разветвляющейся программе необходимо организовать более двух ветвей (многократное ветвление), то можно использовать вложенные условные операторы, то есть в качестве операторов на ветвях *then* или *else* будут также стоять условные операторы.

При этом для удобочитаемости программы необходимо выравнивать по левому краю (то есть располагать на одной вертикальной линии) соответствующую пару ключевых слов *then-else*:

```
If <логическое выражение 1>
  then
    if <логическое выражение 2>
      then <.....>
      else <.....>
    else
      if <логическое выражение 3>
        then <.....>
        else <.....>;
```

*В целом же следует помнить, что по правилам языка Паскаль **каждому else соответствует ближайший стоящий перед ним then (if).***

*Это соглашение позволяет избежать неоднозначности в трактовке выполнения вложенных условных операторов.*

# Примеры

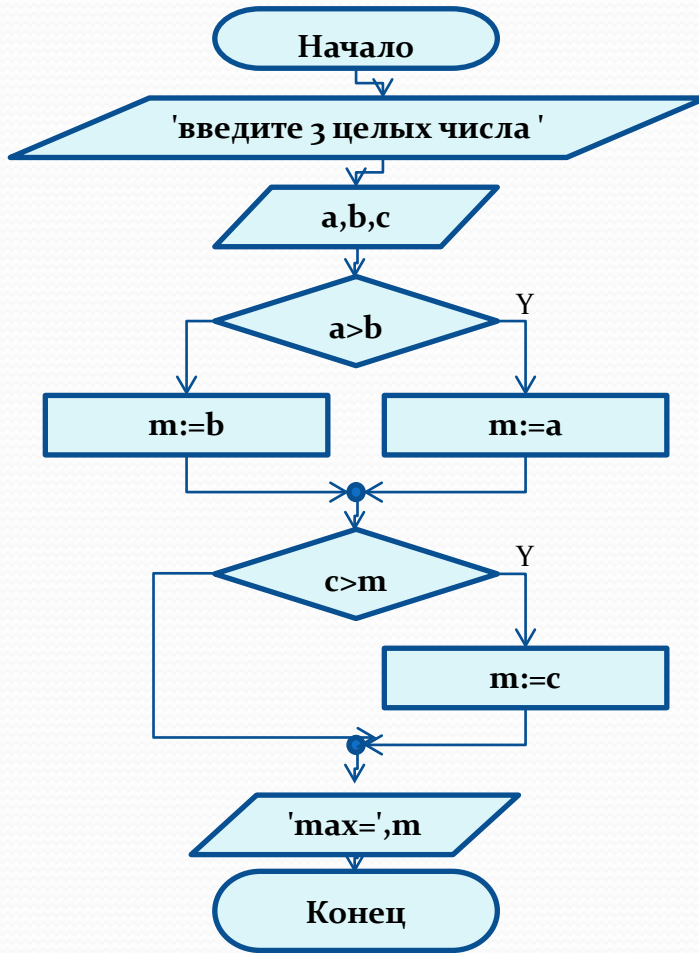
- Пример 1. Даны три целых числа. Найти максимальное из них.

Обозначим исходные числа  $a, b, c$ . Результат  $m$  (максимальное из них)

## 1 способ решения

- Сравним первые два числа, найдем максимальное из них и результат сохраним в переменной  $m$  (полное ветвление).
- Затем сравним  $m$  и  $c$  (неполное ветвление)

# 1 способ решения

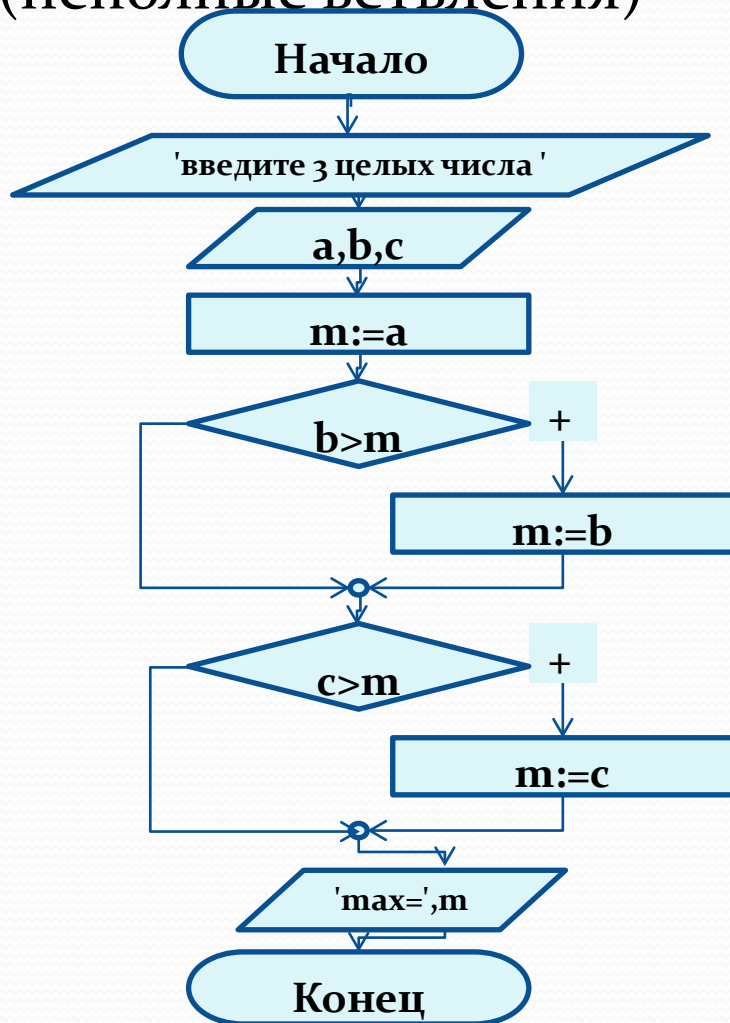


```
max3_sp1.pas
program max3_1sposob;
  var a,b,c,m : integer;
begin
  write('введите 3 целых числа ');
  readln(a,b,c);
  if a>b then m:=a
    else m:=b;
  if c>m then m:=c;
  writeln('max=',m)
end.
```

введите 3 целых числа 2 4 7  
max=7  
введите 3 целых числа 7 4 2  
max=7  
введите 3 целых числа 2 7 4  
max=7

## 2 способ решения

Сначала считаем, что максимальное – первое число ( $m:=a$ ). Потом сравниваем последовательно  $m$  и  $b$ ,  $m$  и  $c$  (неполные ветвления)

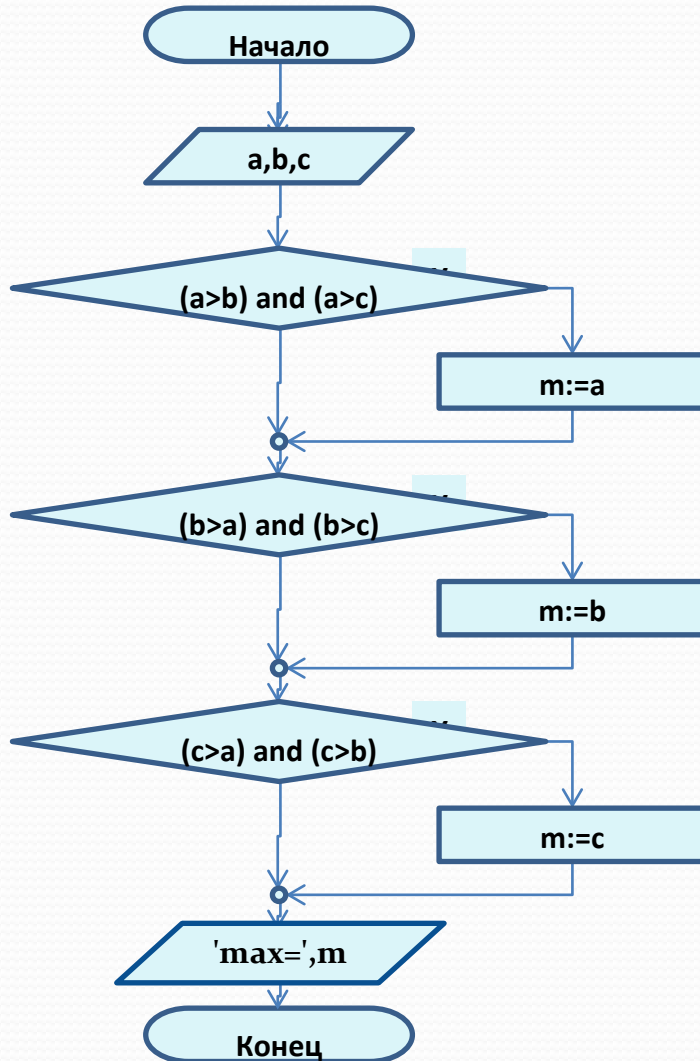


```
program max3_2sposob;
  var a,b,c,m : integer;
begin
  write('введите 3 целых числа ');
  readln(a,b,c);
  m:=a;
  if b>m then m:=b;
  if c>m then m:=c;
  writeln('max=',m)
end.
```

The screenshot shows a window titled 'Pascal ABC' with a menu bar (Файл, Правка, Вид, Программа, Сервис, Помощь) and a toolbar. The code editor contains the Pascal program for finding the maximum of three numbers using the second method. The code is as follows:

### 3 способ решения (этот способ интуитивно понятный, но неэффективный)

Три последовательных неполных ветвления с составными условиями



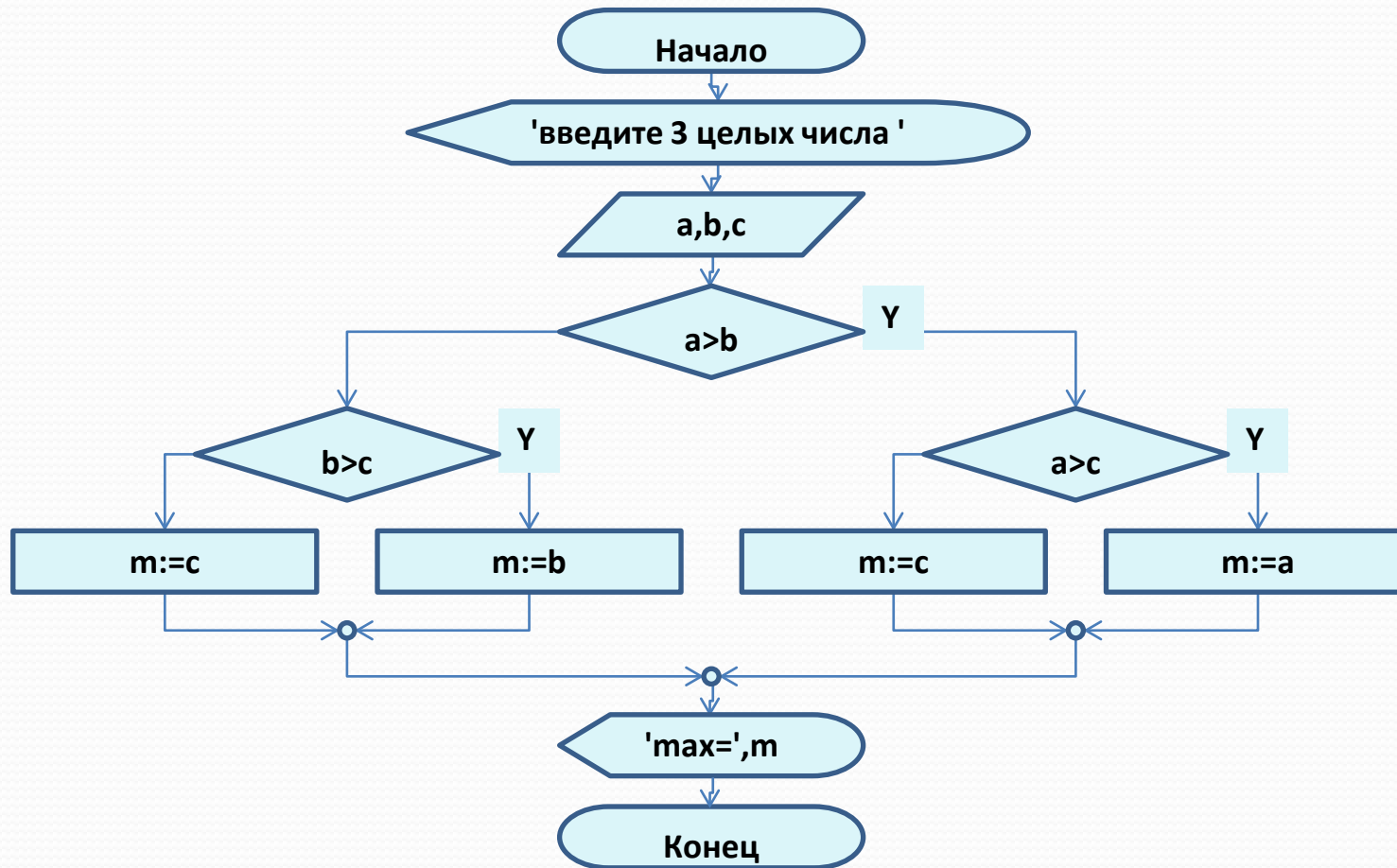
```
Pascal ABC
Файл  Правка  Вид  Программа  Сервис  Помощь
max3_sp1.pas | max3_sp2.pas | max3_sp3.pas

program max3_3способ;
  var a,b,c,m : integer;
begin
  write('введите 3 целых числа ');
  readln(a,b,c);
  if (a>b) and (a>c) then m:=a;
  if (b>a) and (b>c) then m:=b;
  if (c>a) and (c>b) then m:=c;
  writeln('max=',m)
end.
```

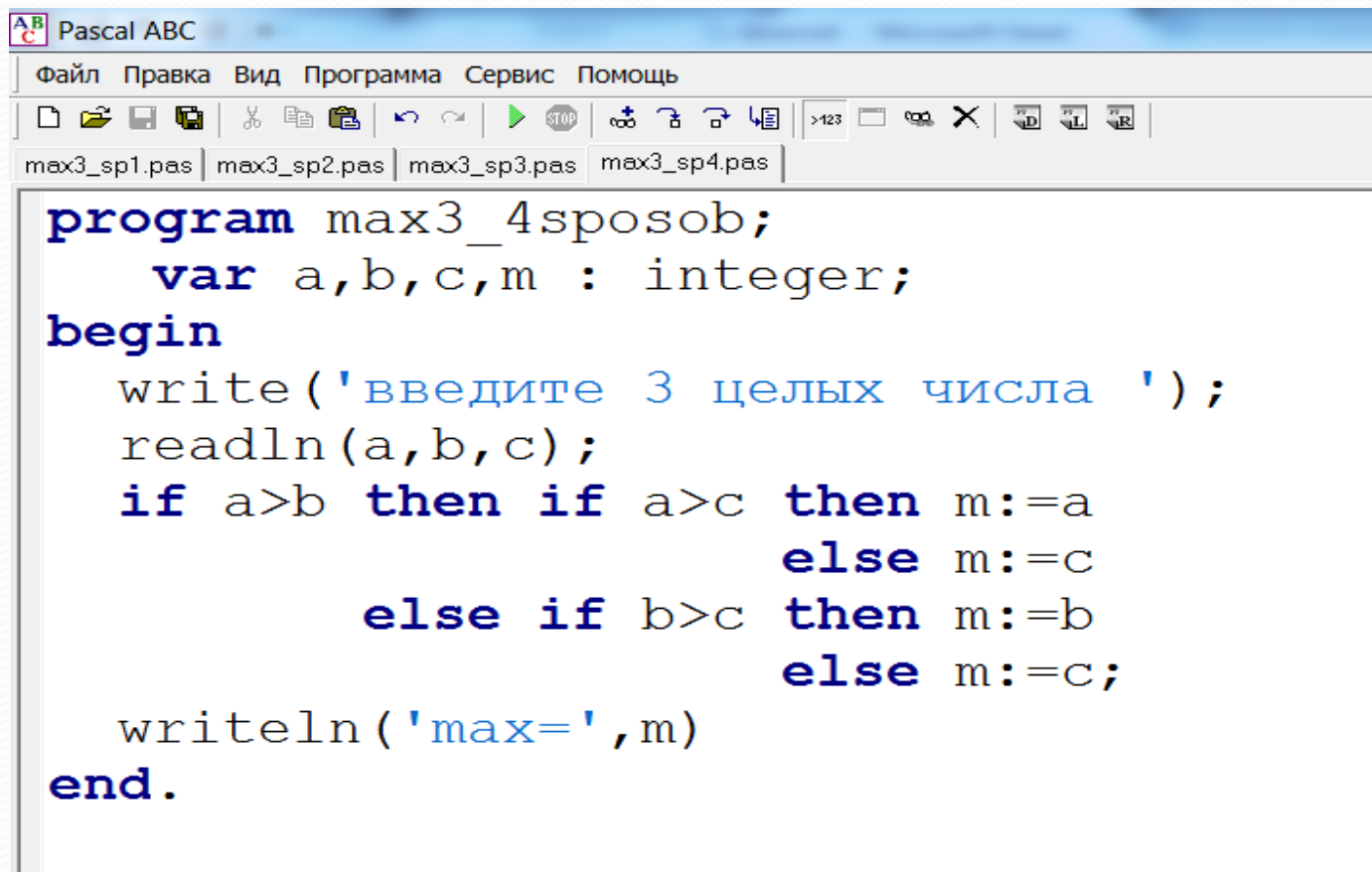


## 4 способ решения

Вложенные ветвления (во внешнем сравниваем два первых числа, а на каждой из ветвей соответственно максимальное из них с третьим)



## 4 способ решения (программа)



```
program max3_4sposob;  
    var a,b,c,m : integer;  
begin  
    write('введите 3 целых числа ');  
    readln(a,b,c);  
    if a>b then if a>c then m:=a  
                else m:=c  
                else if b>c then m:=b  
                else m:=c;  
    writeln('max=',m)  
end.
```

**Пример 2.** Даны три целых числа. Если они расположены в порядке возрастания, то удвоить их. Если второе из них – наибольшее, то возвести в квадрат первое и третье числа, в остальных случаях заменить числа на противоположные по знаку.

Здесь выделяются три различных случая, то есть ветвление будет содержать три ветви: на ветви *else* внешнего условного оператора будет располагаться внутренний полный условный оператор.

```
Program vlogen_vetvlen;
```

```
  var a,b,c : integer;
```

```
begin
```

```
  write('a,b,c='); readln(a,b,c);
```

```
  if (a<b) and (b<c)
```

```
    then
```

```
      begin
```

```
        a:=2*a; b:=2*b; c:=2*c
```

```
      end
```

```
    else if (b>a) and (b>c)
```

```
      then
```

```
        begin
```

```
          a:=a*a; c:=c*c
```

```
        end
```

```
      else
```

```
        begin
```

```
          a:=-a; b:=-b; c:=-c
```

```
        end;
```

```
      writeln(a,b,c)
```

```
end.
```