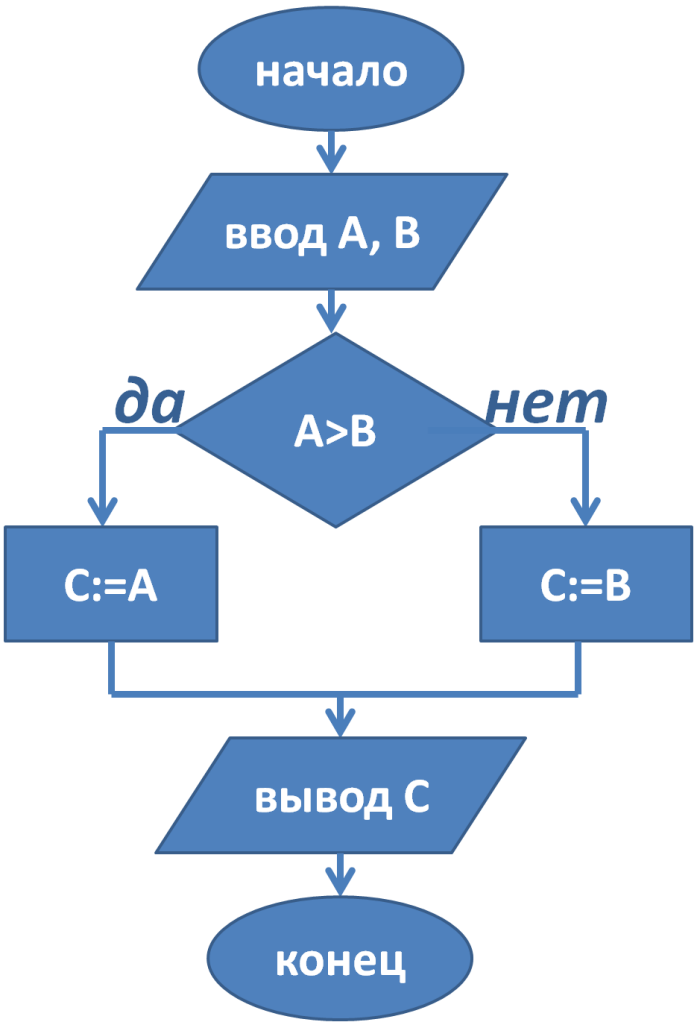
Алгоритмы с ветвящейся структурой

Рассмотрим несколько задач, решение которых на компьютере получается с помощью ветвящихся алгоритмов.

Первая задача: **даны два числа, выбрать наибольшее из них**.

Пусть исходными данными являются переменные A и B. Их значения будут задаваться вводом. Значение большего из них должно быть присвоено переменной C и выведено на экран компьютера. Например, если A=5, B=8, то должно получиться: C=8.

Блок-схема для этого алгоритма (с полным ветвлением):



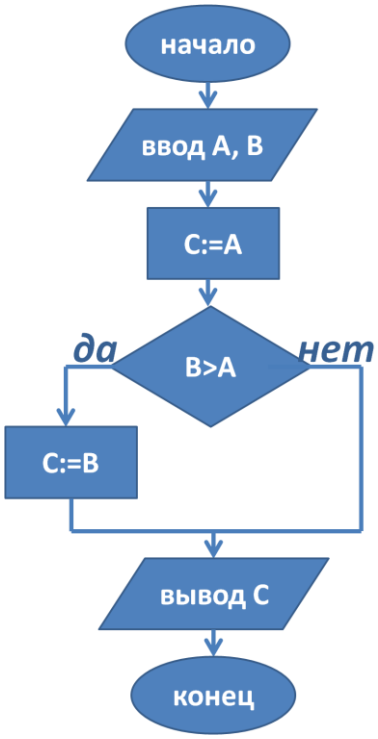
До выполнения на компьютере правильность алгоритма можно проверить путем заполнения **трассировочной таблицы**, в которой последовательно записываются все отдельные операции алгоритма, выполняемые на каждом шаге и значения переменных после выполнения этих операций. Процесс заполнения этой таблицы называется **трассировкой**.

Вот как будет выглядеть трассировка нашего алгоритма для исходных значений A=5, B=8.

Шаг	Операция	A	B	C	Проверка условия
1	Ввод A,B	5	8	-	
2	A>B	5	8	-	5>8, нет (ложь)
3	C:=B	5	8	8	
4	Вывод C	5	8	8	

Эту же самую задачу можно решить , применяя структурную команду неполного ветвления.

Блок-схема такого алгоритма:



Теперь запишем рассмотренные алгоритмы на языках программирования:

1. Для описания переменных нужно указать их тип. Переменные A, B и C числовые величины и они могут принимать любые значения, т.е. являются **вещественными**.

2. В Алгоритмическом языке команда ветвления записывается следующим образом:

если <условие>
  то <действие1>
  иначе <действие2>
кв *(конец ветвления)*

3. В языке Паскаль имеется **оператор ветвления**. Другое его название – **условный оператор**.
Формат полного оператора ветвления следующий:
if <логическое выражение/условие> **then**
<оператор1/действие1>
else
<оператор2/действие2>

Здесь **if** – если, **then** – то, **else** – иначе.

4. Примеры программ с полным ветвлением:
↓*Алгоритмический язык* и *Паскаль*↓

```
алг БОЛЬШЕЕ
нач
вещ A, B, C
  ввод A, B
   если A>B
        то C:=A
        иначе C:=B
   кв
 вывод C
кон
```

```
program bolshee;
var A, B, C: real;
begin
   readln(A,B);
    if A>B
      then C:=A
      else C:=B;
       {конец ветвления обозначает ";"}
    writeln(C)
end.
```

Простой формой логического выражения **является операция отношения.** В Паскале допускаются все виды отношений (ниже указаны их знаки):

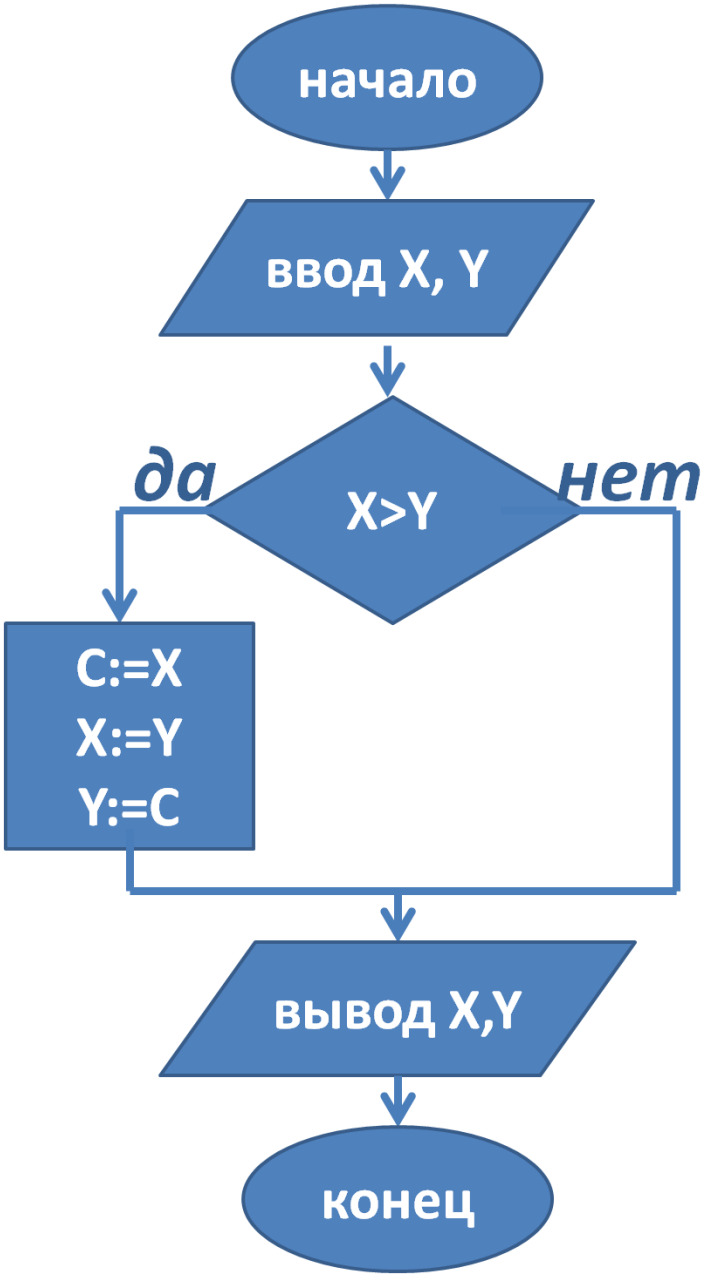
< меньше
> больше
<= меньше или равно
>= больше или равно
= равно
<> не равно

Для неполного ветвления программы будут следующие:

алг
БОЛЬШЕЕ1
нач
вещ A, B, C
  ввод A, B
  C:=A
  если B>A
      то C:=B
  кв
 вывод C
кон

```
program bolshee1;
var A, B, C: real;
begin
  readln(A,B);
  C:=A;
   if B>A
     then C:=B;
          {конец ветвления обозначает ";"}
   write(C)
end.
```

Следующая задача**: упорядочить значения двух переменных X и Y по возрастанию.**
Смысл этой задачи следующий: если для исходных значений переменных справедливо отношение X<=Y, то оставить их без изменения; если же X>Y, то выполнить обмен значениями.
Для обмена нужно использовать третью вспомогательную переменную.



Программы:

```
алг сортировка
нач
вещ X, Y, C
  ввод X, Y
   если X>Y
        то C:=X
           X:=Y
           Y:=C
   кв
  вывод X, Y
кон
```

```
program sortirovka;
var X, Y, C: real;
begin
  readln(X,Y);
    if X>Y
     then begin C:= X;
          X:= Y;
          Y:= C
     end;
    writeln(X,' ', Y)
end.
```

Этот пример иллюстрирует следующее правило Паскаля: *если на какой-то из ветвей оператора ветвления находится несколько последовательных операторов, то их нужно записывать между служебными словами* **begin** и **end**.
Конструкция такого вида:
begin <последовательность операторов> **end**
называется **составным оператором.**

Для того чтобы найти наибольшее среди трех величин A, B, C сначала нужно найти большее из значений A и B и присвоить его какой-то дополнительной переменной, например D; затем найти большее среди D и C. Это значение можно присвоить той же переменной D.

```
program b_iz_treh;
var A, B, C, D: real;
begin
  readln(A,B,C);
    if A>B
     then  D:= A
         else D:=B;
      if C>D
     then  D:= C;
     writeln('наибольшее ',D)
end.
```

(Задания подобного типа попытайтесь решить сами, соответственно своему варианту).

ЛОГИЧЕСКИЕ ОПЕРАЦИИ

Наконец, составим еще один, третий вариант программы определения большего числа из трех.

```
Program ВІТ3;
var A,Б,C,D: real;
begin
 readln(A, B,C);
if     (A>=B)   and  (A>=C)   then D:=A;
if     (B>=A)   and  (B>=C)   then D:=B;
if     (C>=A)   and  (C>=B)   then D:=C;
writeln (D)
end.
```

Нетрудно понять смысл этой программы. Здесь использованы три последовательных неполных ветвления. А условия ветвлений представляют собой *сложные логические* выражения, включающие *логическую операцию* and (И).

Операция and называется логическим умножением или конъюнкцией. Ее результат — «истина», если значения обоих операндов — «истина». Очевидно, что если A >B и A > C, то A имеет наибольшее значение и т. д. В Паскале присутствуют все три основные логические операции:

and — И (конъюнкция),

or — ИЛИ (дизъюнкция),

not — НЕ (отрицание).

Сложные логические выражения

Обратите внимание на то, что отношения, связываемые логическими операциями, заключаются в скобки. ***Так надо делать всегда!*** Например, требуется определить, есть ли среди чисел A, B, C хотя бы одно отрицательное. Эту задачу решает следующий оператор ветвления:

if *(A < 0) o r (B < 0) o r (C < 0)* then write('YES ') else write('NO ');

Выражение, истинное для отрицательного числа, может быть записано еще и так:

not(A>=0).

Обратите внимание на то, что перед else символ «;» не ставится, так как этот символ является разделителем операторов!

Выполните задания под буквами «Г» и «Д» соответствующего варианта.