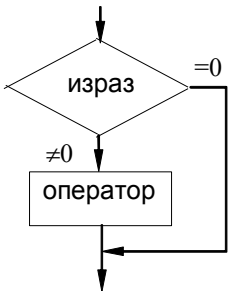
Упражнение №3

ОПЕРАТОРИ ЗА РАЗКЛОНЕНИЯ IF, IF-ELSE

1. Оператор **if**

Операторът **if** се нарича оператор за разклонение с една алтернатива. Само ако **<израз>** има стойност различна от нула (т.е. истина), се изпълнява **<оператор>** и целият оператор завършва. Ако **<израз>** има стойност нула (т.е. лъжа), се изпълнява следващият оператор.

```
if (<израз>)
  <оператор>
```

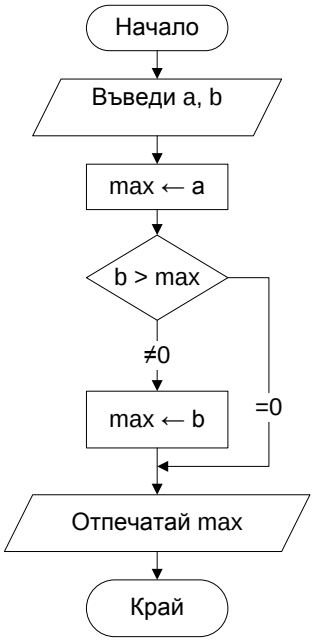


В езика С няма логически тип данни. Ако в израз се използват аритметични операции (**+**, **-**, *****, **/**, **%**, **++**, **--**), той се приема за ИСТИНА, ако стойността му е **РАЗЛИЧНА ОТ НУЛА** и за ЛЪЖА, ако стойността му е **НУЛА**. Ако в израз се използват логически операции (**!**, **&&**, **||**) или операции за отношения (**==**, **!=**, **<**, **<=**, **>**, **>=**), неговата стойност е **1**, ако логическото му значение е ИСТИНА и **0**, ако логическото му значение е ЛЪЖА.

За проверка на отношението „**x** равно на **0**“ се препоръчва да се използва **if(!x)** вместо **if(x==0)**.

За проверка на отношението „**x** различно от **0**“ се препоръчва да се използва **if(x)** вместо **if(x!=0)**.

Пример 1: Напишете програма за определяне на по-голямото от две зададени цели числа **a** и **b**, използвайки представения алгоритъм:

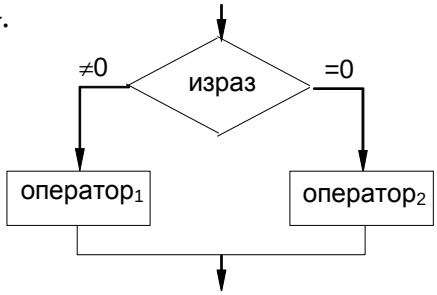



```
// Определяне на по-голямото от две цели числа
// чрез оператор if
#include <stdio.h>
int main()
{
  int a, b, max;
  printf("a = ");
  scanf("%d", &a);
  printf("b = ");
  scanf("%d", &b);

  max = a;
  if(max<b)
    max = b;

  printf("max = %d\n", max);
  return 0;
}
```

2. Оператор **if-else**

Операторът **if-else** се нарича оператор за разклонение с две алтернативи. Ако **<израз>** има стойност различна от нула (т.е. истина), се изпълнява **<оператор$_1$>**, в противен случай се изпълнява **<оператор$_2$>**.

```
if (<израз>)
  <оператор1>
else
  <оператор2>
```



За проверка на отношението „**x** принадлежи на интервала (**a**,**b**)“ се препоръчва да се използва **if(a<x && x<b)**. Езикът C позволява изразът да се запише като **if(a<x<b)** и тъй като редът на изпълнение на операцията „**<**“ е отляво надясно, първо ще се провери изразът **a<x** и резултатът от него (**0** или **1**) ще се провери дали е по-малък от **b**. Компилаторът издава съобщение за несигурна операция. Така следващият код ще отпечата **OK**, което е верен резултат, тъй като 6 принадлежи на интервала (5,10).

```
// Несигурно използване на операциите
float a, b, x;
a = 5; b = 10; x = 6;
if(a<x<b)
  printf("OK\n");
else
  printf("NOT OK\n");
```

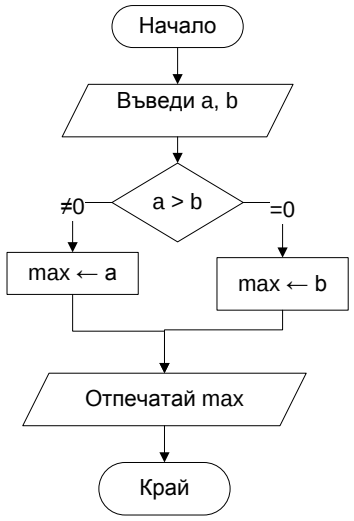
```
// Сигурно използване на операциите
float a, b, x;
a = 5; b = 10; x = 6;
if(a<x && x<b)
  printf("OK\n");
else
  printf("NOT OK\n");
```

Но ако **x = 2**, т.е. извън интервала, кодът ще отпечата отново **OK**, което е грешен резултат. Първо се изчислява изразът 5<6, резултатът е 1 и след това се изчислява изразът 1<10, резултатът също е 1 и затова се отпечатва **OK**.

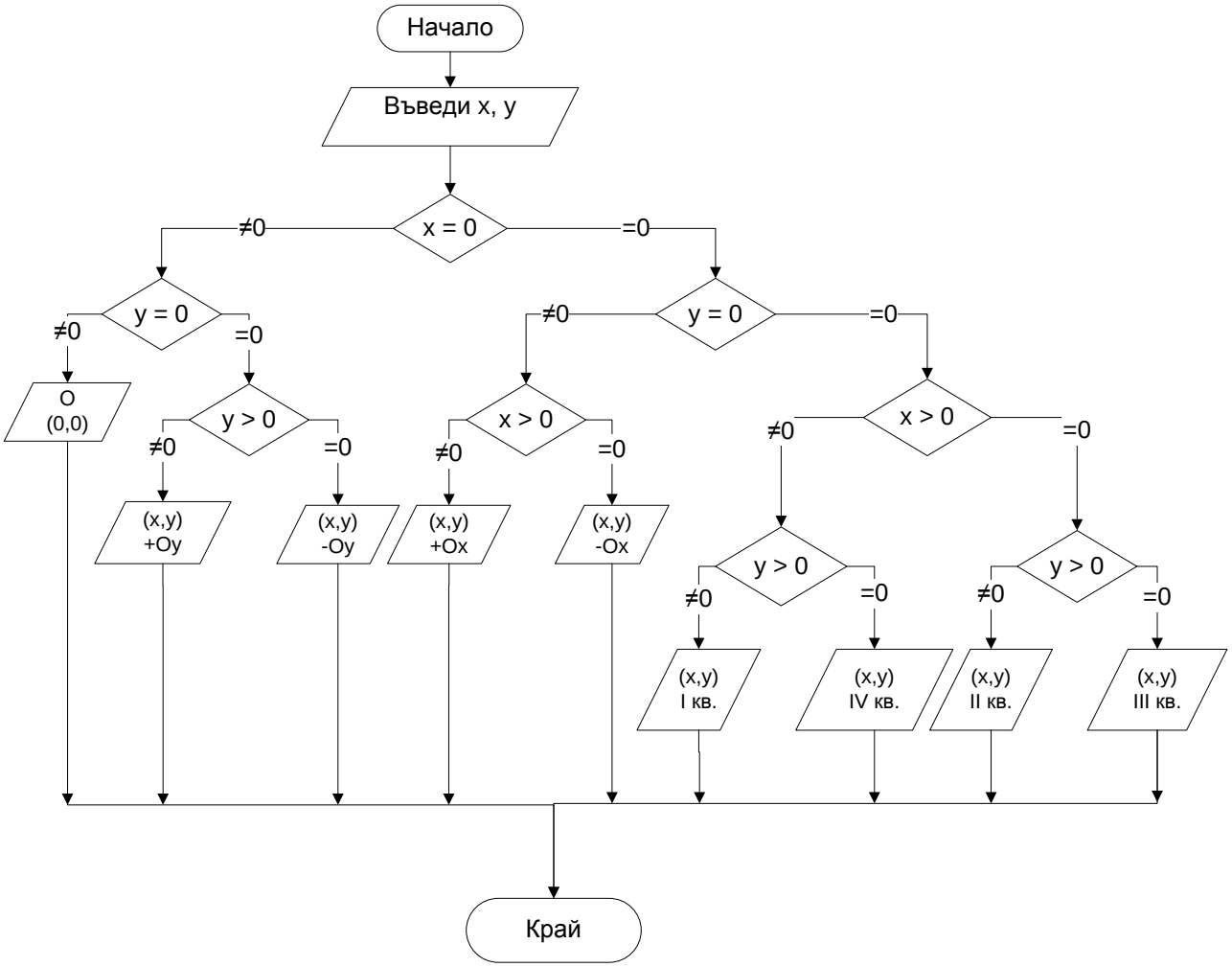
За проверка на отношението „**x** лежи извън интервала (**a**,**b**)“ се препоръчва да се използва **if(a>=x || x>=b)**.

Пример 2: Напишете програмата от Пример 1 за определяне на по-голямото от две зададени цели числа **a** и **b**, използвайки представения алгоритъм:

```
// Определяне на по-голямото от две цели числа
// чрез оператор if-else
#include <stdio.h>
int main()
{
  int a, b;
  printf("a = ");
  scanf("%d", &a);
  printf("b = ");
  scanf("%d", &b);

  if(a>b)
    max = a;
  else
    max = b;

  printf("max = %d\n", max);
  return 0;
}
```



Пример 3: Напишете програма, която въвежда координатите **x** и **y** на точка като цели числа и определя в кой квадрант лежи точката.




```
#include <stdio.h>
int main()
{
  int x, y;
  printf("x = "); scanf("%d", &x);
  printf("y = "); scanf("%d", &y);

  if(!x)                                    // x=0
  {
    if(!y)                                  // x=0, y=0
    {
      printf("(%d,%d) е начало на координатната система\n", x, y);
    }
    else                                    // x=0, y≠0
    {
      if(y>0)                               // x=0, y>0
      {
        printf("(%d,%d) лежи на +Oy\n", x, y);
      }
      else                                  // x=0, y<=
      {
        printf("(%d,%d) лежи на -Oy\n", x, y);
      }
    }
  }
```

```
  else                                  // x≠0
  {
    if(!y)                              // x≠0, y=0
    {
      if(x>0)                           // x>0, y=0
      {
        printf("(%d,%d) лежи на +Ox\n", x, y);
      }
      else                              // x<0, y=0
      {
        printf("(%d,%d) лежи на -Ox\n", x, y);
      }
    }
    else                                // x≠0, y≠0
    {
      if(x>0)                           // x>0, y≠0
      {
        if(y>0)                         // x>0, y>0
        {
          printf("(%d,%d) лежи в I квадрант\n", x, y);
        }
        else                            // x>0, y<0
        {
          printf("(%d,%d) лежи в IV квадрант\n", x, y);
        }
      }
      else                              // x<0, y≠0
      {
        if(y>0)                         // x<0, y>0
        {
          printf("(%d,%d) лежи във II квадрант\n", x, y);
        }
        else                            // x<0, y<0
        {
          printf("(%d,%d) лежи в III квадрант\n", x, y);
        }
      }
    }
  }
  return 0;
}
```