ТЕМА 4.

Управляющие конструкции языка С

Условный оператор

Разветвляющиеся алгоритмы

Задача. Ввести два целых числа и вывести на экран наибольшее из них.

Идея решения: надо вывести на экран первое число, если оно больше второго, или второе, если оно больше первого.

Особенность: действия исполнителя зависят от некоторых условий (*если … иначе …*).

Алгоритмы, в которых последовательность шагов зависит от выполнения некоторых условий, называются разветвляющимися.

Организация ветвящихся процессов: оператор if

а) в полной форме:

```
if (выражение)
        оператор_1;
else
        оператор_2;
```

б) в сокращенной форме:

```
if  (выражение)
        оператор_1;

        {
        оператор_1;
        оператор_2;
        ……
        оператор_n;
        }.
```

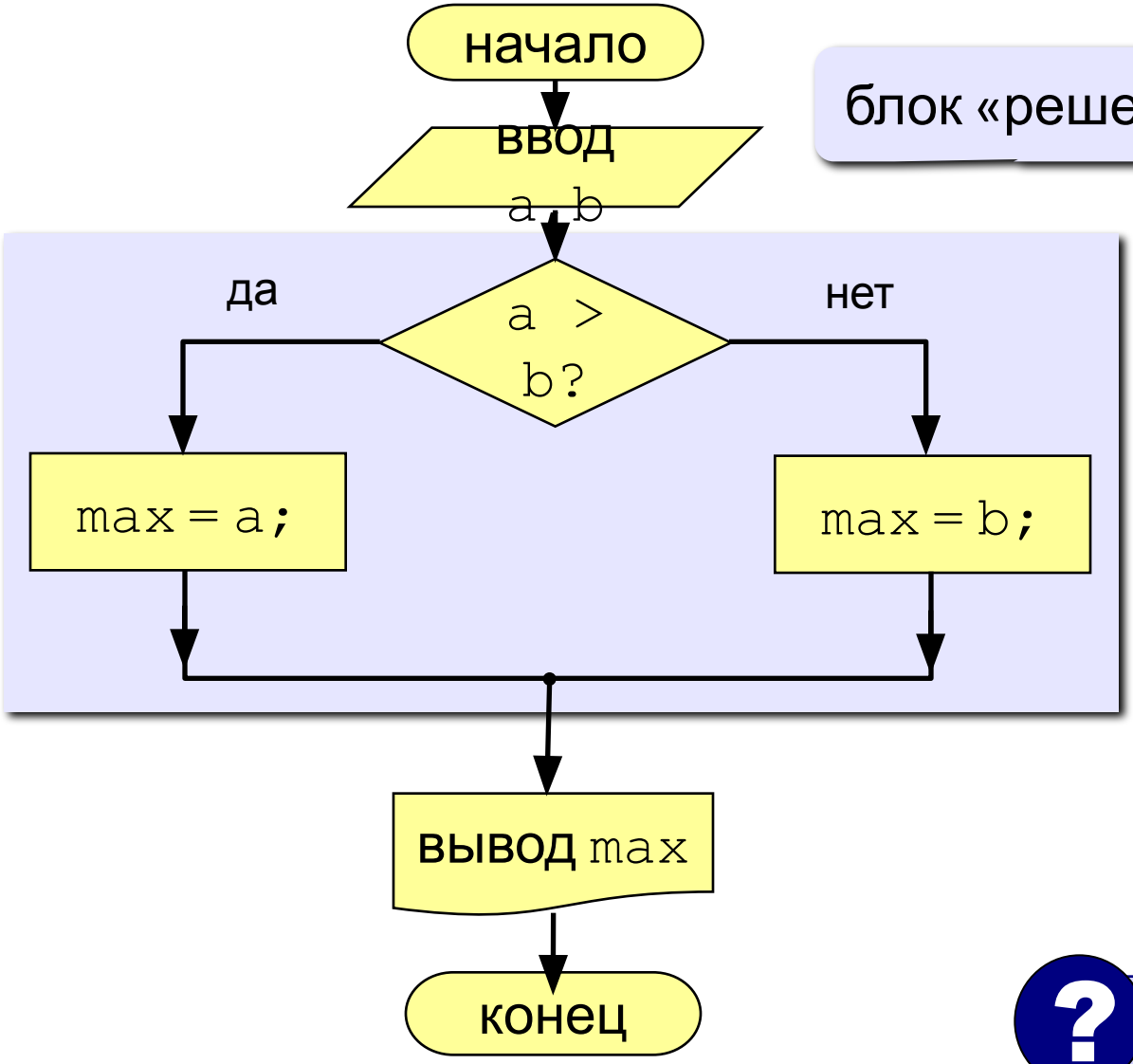
Условный оператор

```
if ( условие )
   {
       // что делать, если условие верно
   }
else
   {
       // что делать, если условие неверно
   }
```

Особенности:

- вторая часть (*else* …) может отсутствовать (неполная форма)
- если в блоке один оператор, можно убрать { }

Вариант 1. Блок-схема



блок «решение»

полная форма ветвления

? Если a = b?

Вариант 1. Программа

```
main()
{
    int a, b, max;
    printf("Введите два целых числа\n");
    scanf("%d%d", &a, &b );
    if (a > b) {
        max = a;
        }
    else {
        max = b;
        }
    printf("Наибольшее число %d", max);
 }
```

полная форма условного оператора

Что неправильно?

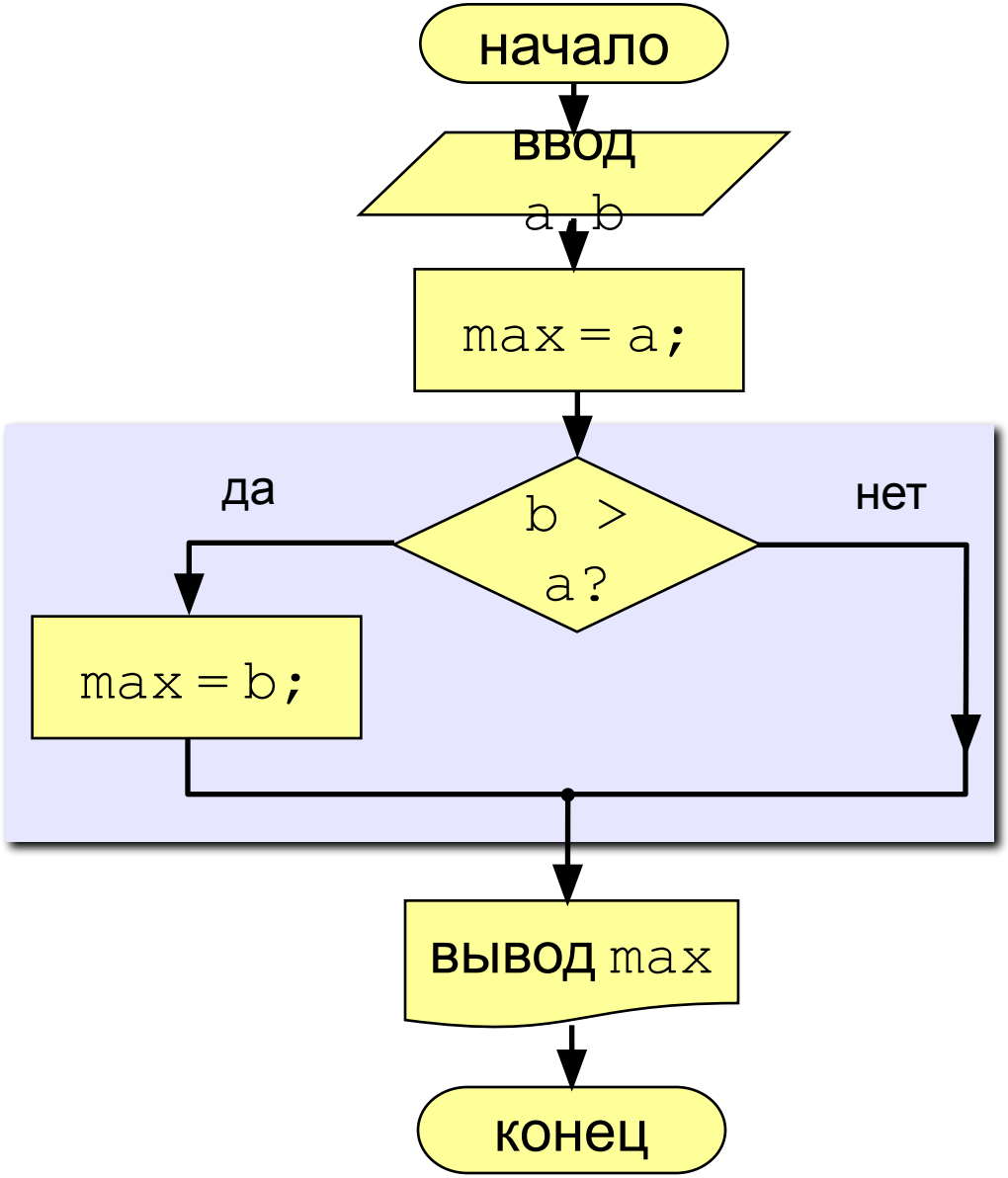
```
if ( a > b ) {
   a = b;
   }
else
   b = a;
```

```
if ( a > b ) {
   a = b; }
else
   b = a;
```

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

```
if ( a > b ) {
   a = b;
   c = 2*a; }
else
   b = a;
```

Вариант 2. Блок-схема



неполная форма ветвления

Вариант 2. Программа

```
main()
{
   int a, b, max;
   printf("Введите два целых числа\n");
   scanf("%d%d", &a, &b );
   max = a;
   if (b > a)
     max = b;
   printf("Наибольшее число %d", max);
}
```

неполная форма условного оператора

Вариант 2Б. Программа

```
main()
{
   int a, b, max;
   printf("Введите два целых числа\n");
   scanf("%d%d", &a, &b );
   max = b;
   if ( a > b )
      max = a;
   printf("Наибольшее число %d", max);
}
```

Что неправильно?

```
if a > b
      a = b;
else b = a;
```

```
if a > b {
   a = b;
}
else b = a;
```

```
if a > b
      a = b;
else b = a;
```

Пример 1:

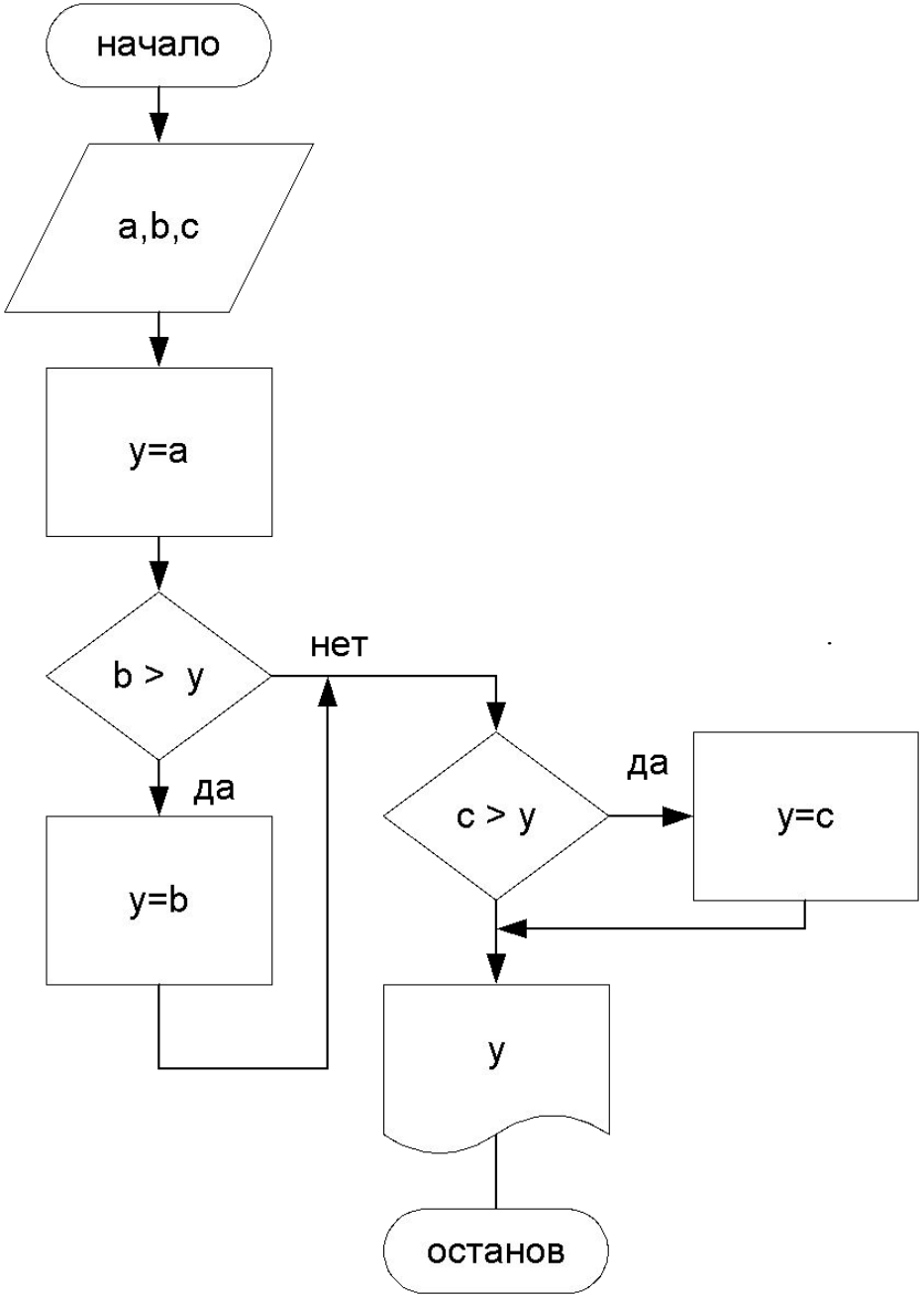
```
# include <conio.h>
 void main ( void)
{
  clrscr ();
  float a, b, rez ;
  printf (“Введите значение a и b: “);
  scanf (“%f  %f “, &a, &b);
  if (b= =0)
          printf (“Отношение a / b не определено \n “);
   else
        {
          rez = a / b;
          printf (“Отношение a / b равно %6, 3f\n”, rez);
        }
  getch( );
  }
```

равно ==

Не равно !=

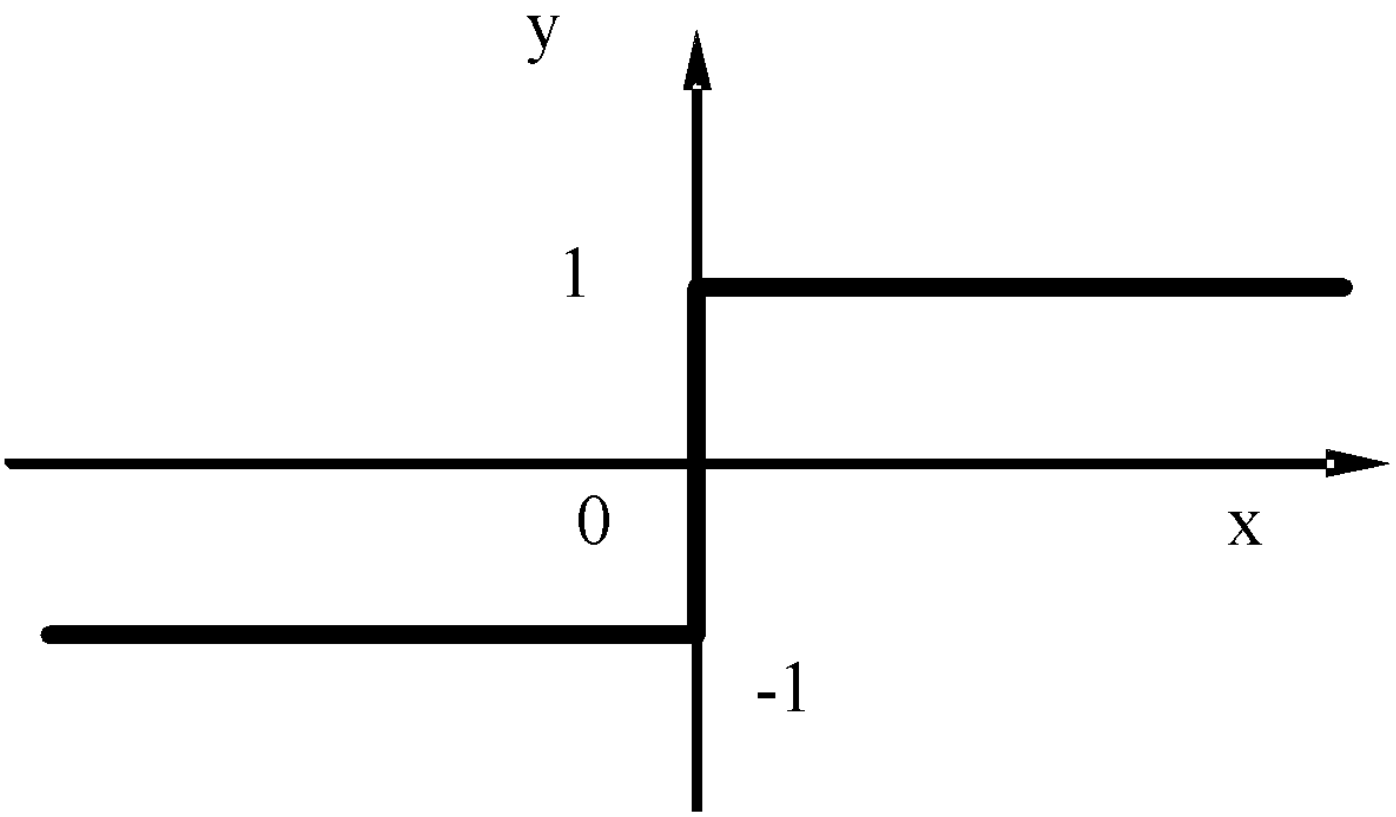
if (c=ch,ch=getch( )).

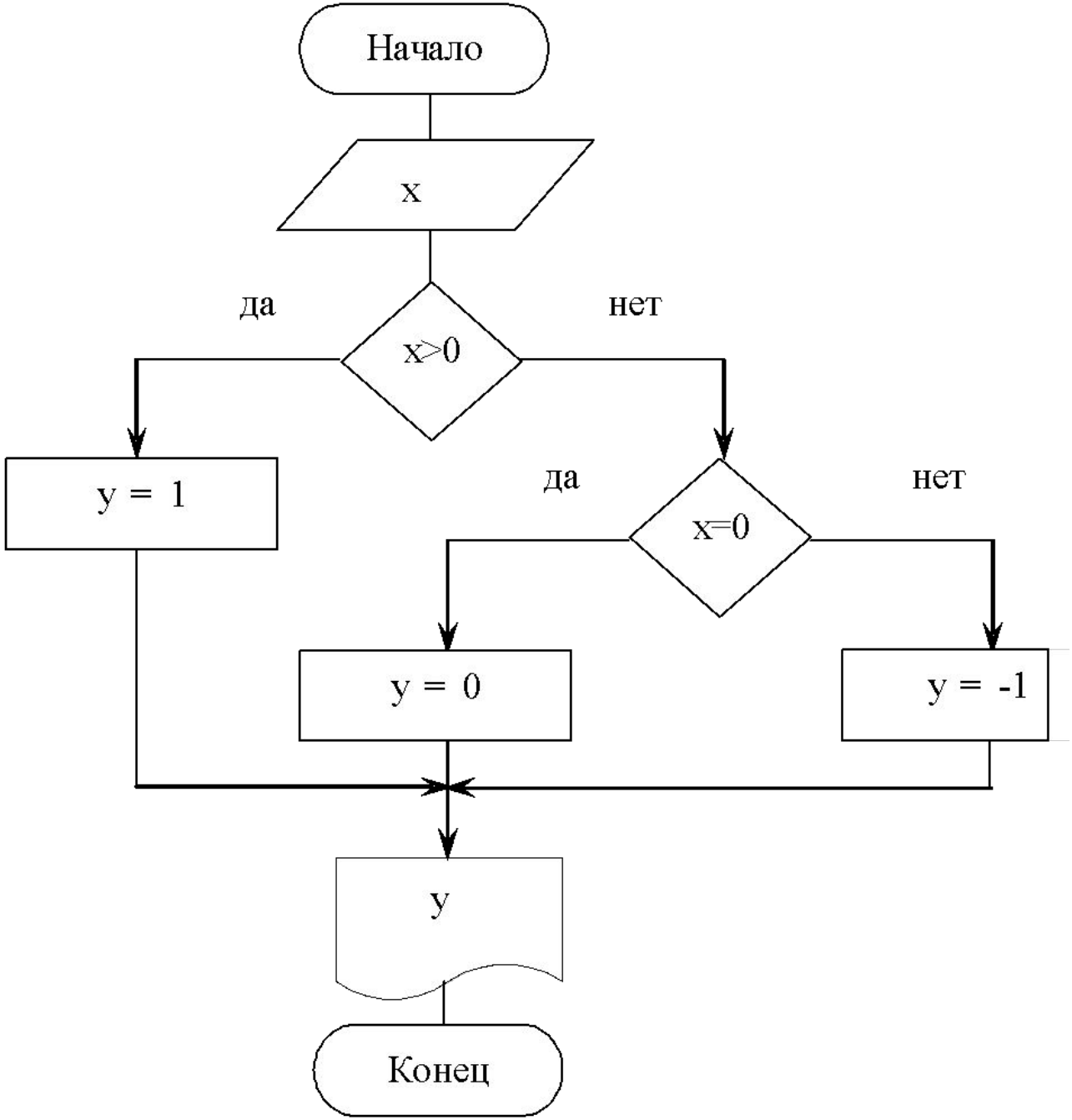
Пример 3. $y = \max(a, b, c)$




```
main();
  {
Float y,a,b,c;
Ввод a,b,c
   y=a;
   If b>y
            y=b;
   If c>y
            y=c;
   Печать  y
}
```

***Пример 2*: Программа функции sqn(x). Она вычисляет знак введенного числа x, т.е. sqn(x) принимает значение 1, если x > 0, значение -1, если x < 0, и значение 0, если x=0.**




Начало
x
да
нет
x>0
y = 1
да
нет
x=0
y = 0
y = -1
y
Конец

```
#include  <stdio.h>
main()
{
int sqn;
float  x;
printf (“Введите число”);
scanf (“% f “, &x);
if  (x>0)
  {  sqn =1; printf (“число %f  положительное sqn = %d \n “,x,
sqn);}
if (x= =0)
  { sqn=0; printf (“число %f равно нулю sqn= %d \n”, x, sqn);}
if (x<0)
   { sqn = -1; printf (“число %f отрицательное sqn= %d \n”, x,
sqn);}
}
```

```
if (условие) оператор;
else if(условие) оператор;
        else if((условие)оператор;
                    …..
           else оператор;
```

```
……..
 if (x>0)
        { sqn = 1; printf (“число %f  положительное \n”, x); }
  else
        if (x<0)
             { sqn= -1; printf (“число %f отрицательное \n”, x);
 }
        else
             { sqn =0; printf (“число %f равно нулю \n”, x); }
```