

# Информатика: Основы программирования

---



## Язык Си

---

Кафедра О7  
Лектор Татьяна Ильинична Лазарева



Лекция №5  
Тема:  
Циклы.



# Циклы

---

**Цикл** – это **многократное** выполнение одинаковых действий.

По расположению условия:  
цикл с **предусловием**  
цикл с **постусловием**

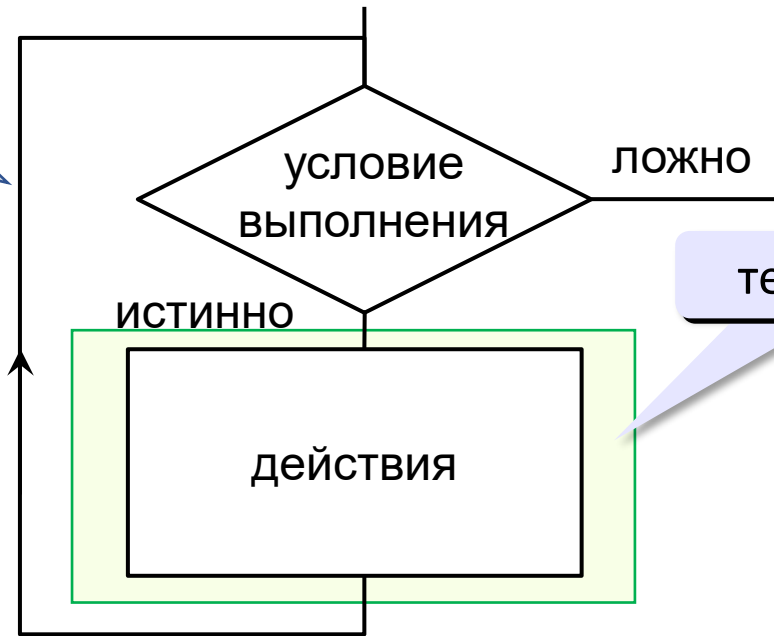
По числу повторов:  
цикл с **известным числом** шагов  
(арифметический);  
цикл с **неизвестным числом** шагов  
(цикл с условием, итерационный).



В Си все циклы итерационные

# Цикл с предусловием

Схема  
поясняет  
работу  
цикла



тело цикла

имя цикла  
условие

действия

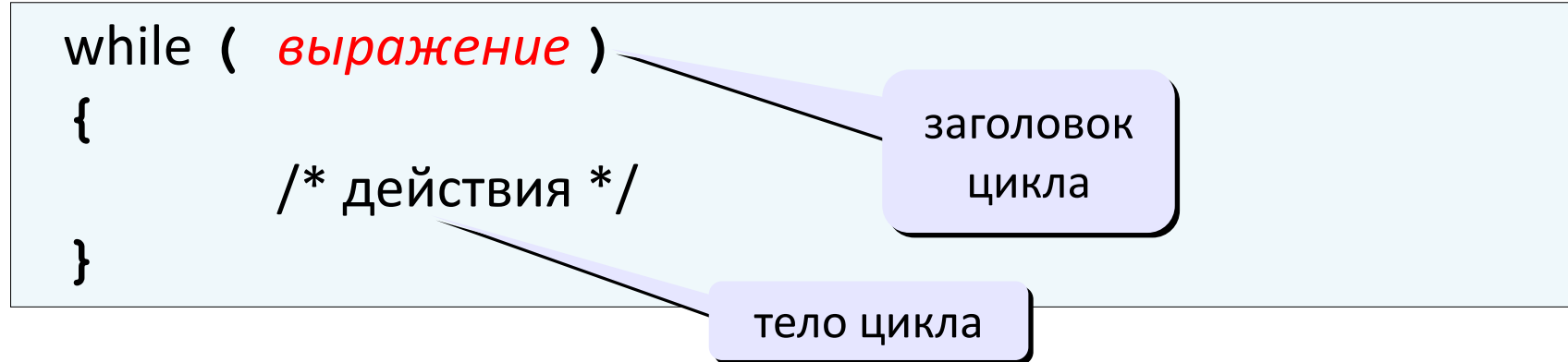
имя цикла

Схема  
алгоритма,  
выполнена  
по ГОСТу

```
while ( выражение )  
    оператор;
```

```
for ( выражение 1 ; выражение 2; выражение 3 )  
    оператор;
```

# Оператор **while**



Если значение выражения **ИСТИННО** (не равно 0), то выполняется тело цикла. Когда значение выражения становится **ЛОЖНО**, цикл заканчивается.



В языке Си **любое число**, не равное **нулю** (2, -1, 0.25), обозначает **истину**, а ноль — ложь.



- **выражение** пересчитывается каждый раз при входе в цикл
- если значение **выражения** на входе в цикл **равно нулю**, тело цикла не выполняется ни разу
- если тело цикла – один оператор, то { } не нужны

# Сколько раз выполнится цикл?

```
a = 4; b = 6;  
while ( a < b ) a ++;
```

2 раза  
a = 6

```
a = 4; b = 6;  
while ( a < b ) a += b;
```

1 раз  
a = 10

```
a = 4; b = 6;  
while ( a > b ) a ++;
```

0 раз  
a = 4

```
a = 4; b = 6;  
while ( a < b ) b = a - b;
```

1 раз  
b = -2

```
a = 4; b = 6;  
while ( a < b ) a --;
```

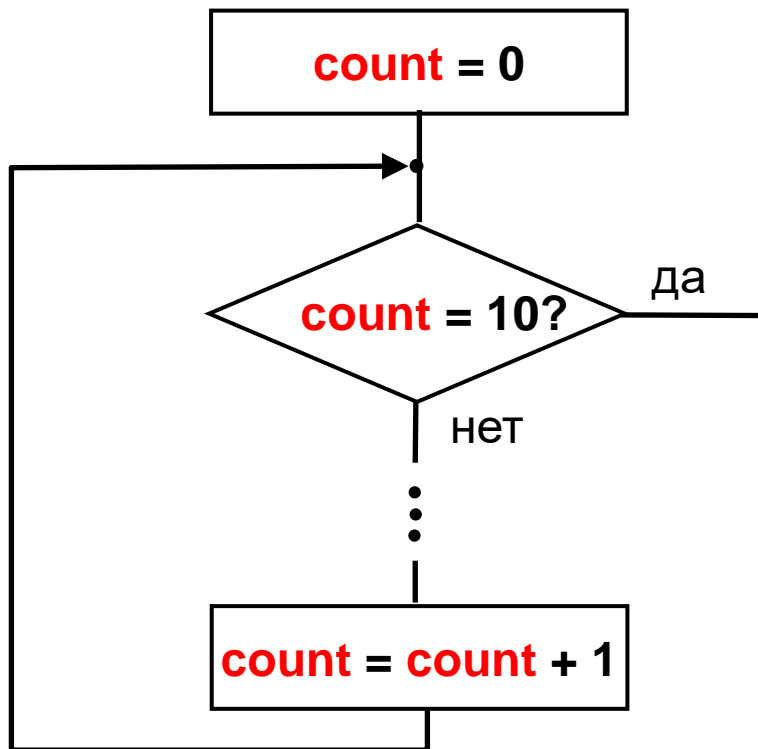
зависит от типа  
переменной a

# Арифметический цикл

**Задача.** С клавиатуры вводятся 10 чисел.  
Найти среднее арифметическое этих чисел.

**Алгоритм.** Считывать значения по одному и прибавлять к общей сумме.

**Проблема.** Как считать число повторов?  
Нужен счетчик. Переменная **count** будет считать шаги.



А  
Л  
Г  
О  
Р  
И  
Т  
М

# Текст программы с использованием оператора *while*

```
#include <stdio.h>
#include <locale.h>
main()
{
    setlocale(LC_ALL,"Russian");
    double number, sum=0;
    int count=0;
    printf ( "\n Введите числа\n " );
    while ( count !=10 )
    {
        scanf ( "%lf", &number )
        sum+=number;
        count ++;
    }
    printf ("Среднее а.р.= % lf \n", sum/10 );
    return 0;
}
```

Начальное значение суммы

Начальное значение  
счетчика повторов

Условие выполнения цикла

Приращение счетчика



```

#include <stdio.h>
#include <locale.h>
main()
{
    setlocale(LC_ALL,"Russian");
    double number, sum=0;
    int count=0;
    printf ( "\n Введите числа\n " );
    while ( count !=10 )
    {
        scanf ( "%lf", &number );
        sum+=number;
        count ++;
    }
    printf ("Среднее ар.= % lf \n", sum/10 );
    return 0;
}

```

Тест 1

```

Введите 10 чисел
2 2 2 2 2 2 2 2 2 2
Среднее ар.= 2,000000

```

Тест 2

```

Введите 10 чисел
1 2 3 1 2 3 1 2 3 1
Среднее ар.= 1,900000
-----

```

Тест 3

printf ( " Среднее ар.= %5.1lf \n", sum/10 );

```

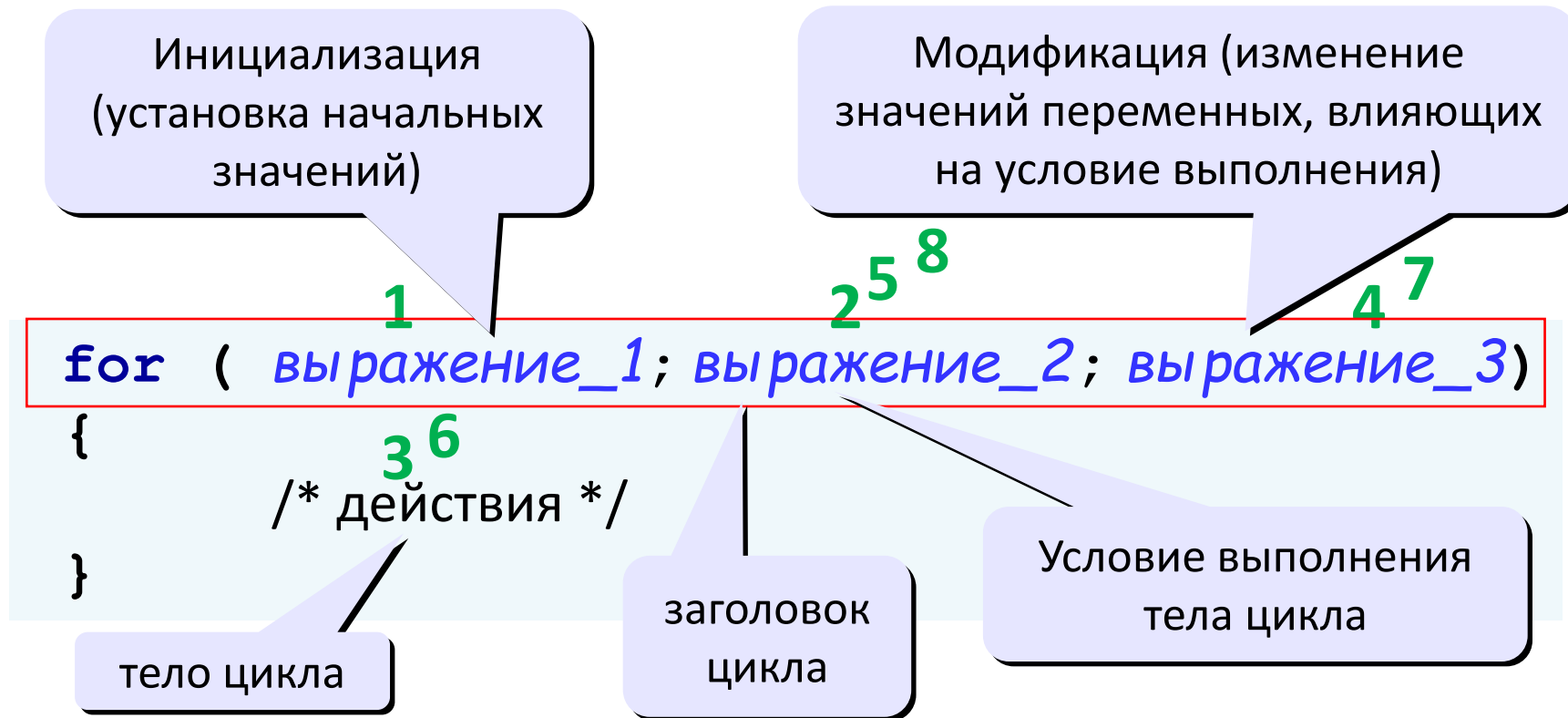
Введите 10 чисел
1 2 3 1 2 3 1 2 3 1
Среднее ар.=      1,9

```

```
while ( 1 )    // бесконечный цикл  
{тело цикла}
```

```
while ( 0 )  
{ ... }      // цикл не выполнится ни разу
```

# Оператор цикла *for*



# Оператор *for*

---

## Особенности:

- *for* – инструкция **цикла с предусловием**
- *выражения* могут быть **любыми**
- *выражение\_1* вычисляется только один раз
- *выражение\_2* пересчитывается каждый раз при входе в цикл
- если значение *выражения\_2* на входе в цикл равно нулю, тело цикла не выполняется ни разу
- если тело цикла – одна инструкция, { } не нужны

# Текст программы с оператором *for*

```
#include <stdio.h>
#include <locale.h>
main()
{
    setlocale(LC_ALL,"Russian");
    double number,sum=0;
    int count=0;
    printf ( "\n Введите 10 чисел\n " );
    for ( count=0; count !=10; count++ )
    {
        scanf ( "%lf", &number );
        sum+=number;
    }
    printf ( " Среднее ar.= %5.1lf\n", sum/10 );
    return 0;
}
```

Начальное значение  
счетчика повторов

Приращение счетчика  
(модификация)

Условие выполнения цикла

```
#include <stdio.h>
#include <locale.h>
main()
{
    setlocale(LC_ALL,"Russian");
    double number,sum=0;
    int count=0;
    printf ( "\n Введите 10 чисел\n " );
    for ( count=0; count !=10; count++ )
    {
        scanf ( "%lf", &number );
        sum+=number;
    }
    printf ( " Среднее ар.= %5.1lf \n",
sum/10 );
    return 0;
}
```

```
Введите 10 чисел
2 2 2 2 2 2 2 2 2 2
Среднее ар.= 2,0
```

```
Введите 10 чисел
1 2 3 1 2 3 1 2 3 1
Среднее ар.= 1,9
```

```
Введите 10 чисел
2 2 2 2 2 2 2 2 2 2
Среднее ар.= 2,0
```

# Оператор цикла *for*\_Особенности:

любое из *выражений* в заголовке может отсутствовать,  
; ставятся всегда

```
#include <stdio.h> main()  
#include <locale.h>  
  
{  
    setlocale(LC_ALL,"Russian");  
    double number,sum=0;  
    int count=0;  
    printf ( "\nВведите 10 чисел\n " );  
    for ( ; count !=10; )  
    {  
        scanf ( "%lf", &number );  
        sum+=number;  
        count++ ;  
    }  
    printf ( " Среднее ар.= %5.1lf \n", sum/10 );  
    return 0;  
}
```

Начальное значение  
счетчика повторов

Инициализация  
не требуется

Модификация не  
требуется

Счетчик изменяется в  
теле цикла

## Результат к предыдущему слайду

```
Введите 10 чисел  
1 2 3 1 2 3 1 2 3 1  
Среднее арифметическое = 1,9
```

Еще одна особенность

for ( a = 5 ; ; a += 2 )  
оператор;

Отсутствие выражения равного (==) «ИСТИНА»  
дает **бесконечный цикл**.



**Пример.** С клавиатуры вводят 15 чисел.  
Найти количество чисел, принадлежащих  
отрезку  $[-5;5]$ .

```
#include <stdio.h>
int main (void)
{
    float a, int kol = 0, i;
    for (i=1; i≤15; i++)
    {
        scanf ("%f ", &a);
        if (a ≤ 5 && a ≥ -5)
            kol++;
    }
    printf ("kol=%d",kol);
    return 0;
}
```

## Вычислить $n!$ факториал

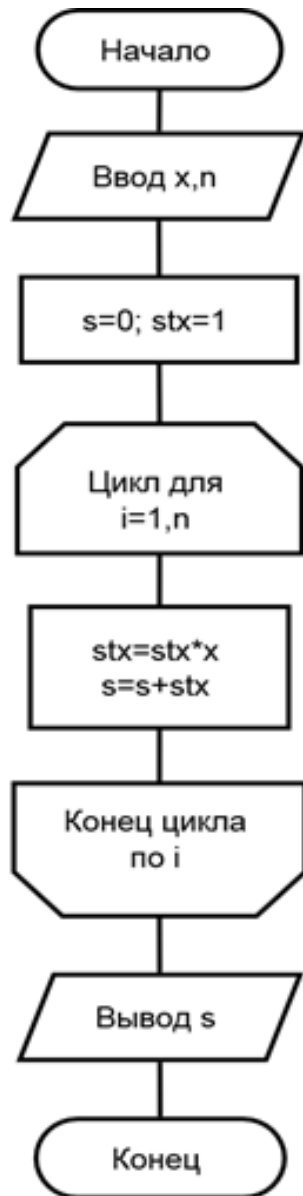
```
#include <stdio.h>
int main (void)
{
    unsigned int n, i, f=1;
    printf ("Введите n\n");
    scanf ("%u", &n);
    for (i=2; i<=n; i++)
        f*=i;
    printf ("%u! = %u", n, f);
    return 0;
}
```

## Вычислить $x^n$

```
#include <stdio.h>
int main (void)
{
    int n, i;
    double x,stx;
    printf ("n=");
    scanf ("%d", &n);
    printf ("x=");
    scanf ("%lf", &x);
    stx=x;
    for (i=2; i<=n; i++)
        stx*=x;
    printf (" stx =%7.3lf\n", stx );
    return 0;
}
```

```
n=5
x=2
stx = 32.000
```

**Пример.** Дано вещественное число  $x$  и натуральное  $n$ . Вычислить значение суммы  $s = x + x^2 + x^3 + \dots + x^n$



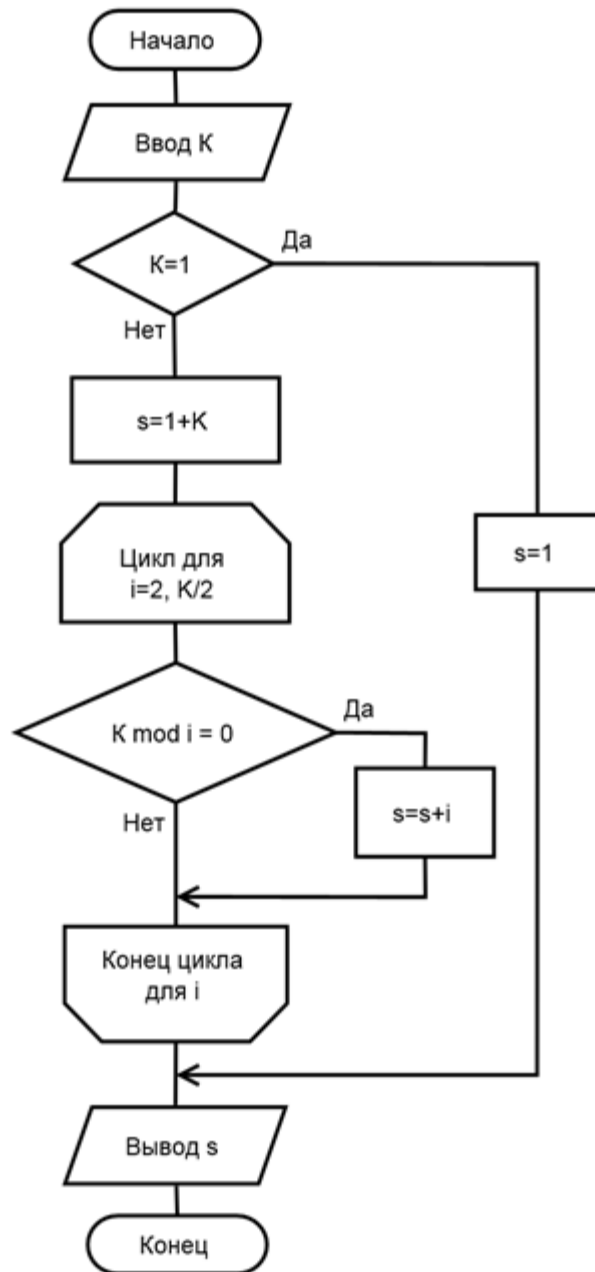
```
#include <stdio.h>
int main (void)
{
    int n, i;
    double x,s,stx;
    printf ("n=");
    scanf ("%d", &n);
    printf ("x=");
    scanf ("%lf", &x);
    s=0;
    stx=1;
    for (i=1; i<=n; i++)
    {
        stx*=x;
        s+=stx;
    }
    printf ("s=%7.3lf\n", s);
    return 0;
}
```

Ручное тестирование:  
при  $x=2$ ,  $n=3$   
 $2+4+8=14$

тест

```
n=3
x=2
s= 14.000
```

**Пример.** Дано натуральное число. Определить сумму его делителей.



```
#include <stdio.h>
int main (void)
{
    int k,s,i;
    printf ("k=");
    scanf ("%d", &k);
    if (k==1)
        s=1;
    else
    {
        s=1+k;
        for (i=2; i<=k/2; i++)
            if (k%i==0)
                s+=i;
    }
    printf ("s=%d\n", s);
    return 0;
}
```

```
for (i=2; i<=k/2, k%i==0 ; i++)
    s+=i;
```

Тест 1

```
k=1
s=1
```

Тест 2

```
k=4
s=7
```

Тест 3

```
k=17
s=18
```

```
k=4
s=7
```

### Пример.

Дано целое число. Определить сумму цифр в заданном числе.

**Алгоритм:** число N делится на 10 и отбрасывается остаток, и так до тех пор, пока результат деления не станет равен нулю.

С помощью переменной count (она называется счетчиком) считаем, сколько раз выполнялось деление – столько цифр и было в числе.

**Исходные данные:** переменная N – целое число.

**Результат:** переменная count – количество цифр в числе N.

### Текст программы

```
#include <stdio.h>
#include <locale.h>
main()
{ setlocale(LC_ALL,"Russian");
  int N, count=0;
  printf ( "\n Введите число N: " );
  scanf ( "%d", &N );
  while ( N > 0 )
  {
    N /= 10;  // отсекаем последнюю цифру
    count ++; // увеличиваем счетчик цифр
  }
  printf ( "В этом числе %d цифр\n", count );
}
```

7215/10	N=721	count =1
721/10	N=72	count =2
72/10	N=7	count =3
7/10	N=0	count =4

Результат

Введите число N: 7215  
В этом числе 4 цифр

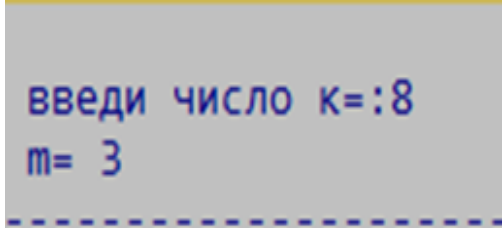
### Пример.

Найти такое значение  $m$ , при котором  $2^m > k$ .

$k$  вводится с клавиатуры.

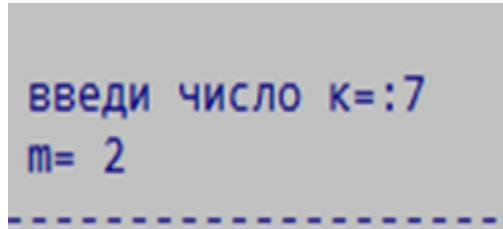
```
#include <stdio.h>
#include <locale.h>
int main (void)
{
    setlocale(LC_ALL,"Russian");
    int m=0, st2=2, k;
    printf ( "\n введи число k=" );
    scanf("%d", &k);
    while (st2 <= k)
    {
        m++;
        st2 *= 2; // Вычисление степени
    }
    printf (" m= %d ", m);
    return 0;
}
```

Тест 1



```
введи число k=:8
m= 3
```

Тест 2



```
введи число k=:7
m= 2
```

При отсутствии `выр1`, или `выр3` считается, что их просто нет в конструкции цикла; при отсутствии `выр2`, предполагается, что его значение как бы всегда истинно и цикл становится бесконечным. Например,

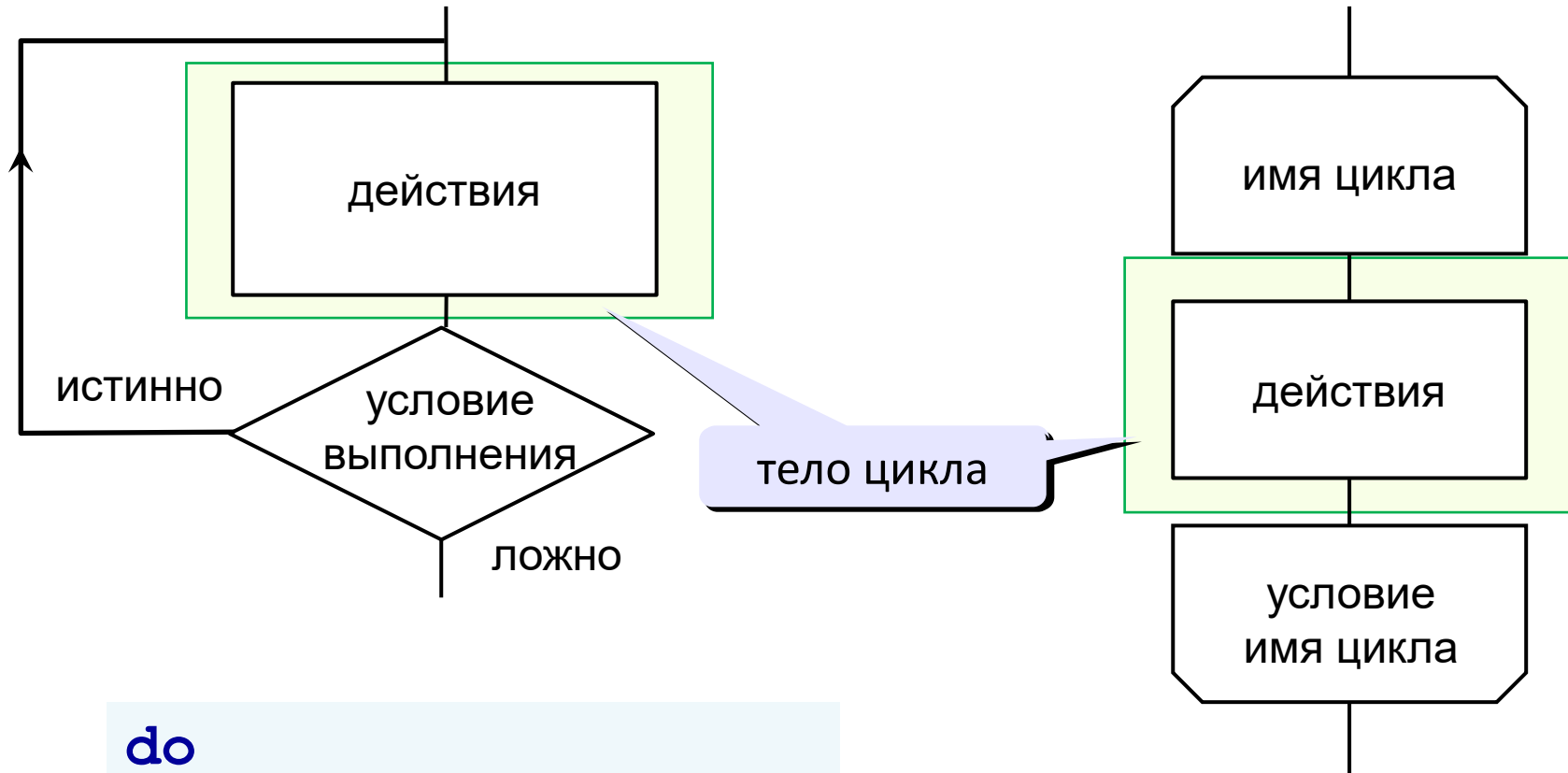
```
for (;;)
{
    ...
}
```

есть "бесконечный" цикл, выполнение которого, вероятно, прерывается каким-то другим способом, например с помощью инструкций `break` или `return`.

Оператору `for` эквивалентна следующая конструкции:

```
выражение 1;
while (выражение 2)
{
    оператор;
    выражение 3;
}
```

# Цикл с постусловием



```
do  
    инструкция;  
while ( выражение );
```



Хотя бы один раз тело цикла обязательно выполнится



# Инструкция *do ... while*

---

```
do
{
    /* действия */
}
while ( выражение, определяющее условие );
```



## Особенности:

- *выражение* может быть **любым**
- *выражение* определяет условие возврата к началу цикла (условие повторения цикла)
- *выражение* пересчитывается каждый раз при выходе из цикла
- тело цикла выполняется хотя бы один раз
- если тело цикла – одна инструкция, { } не нужны

# Цикл с постусловием

---

Чаще всего цикл с постусловием применяется для **проверки** корректности **ввода**

**Задача:** Ввести целое **положительное** число.

**Проблема:** Как не дать ввести отрицательное число или ноль?

**Решение:** Если вводится неверное число, вернуться назад к вводу данных (цикл!).

```
int num, count = 0;
printf ("Введите целое положительное число\n");
do
    scanf ("%d", &num);
while ( num <= 0 );
```

### Пример.

Ввести натуральное число и найти сумму его цифр.

Организовать ввод числа так, чтобы нельзя было ввести отрицательное число или нуль.

Любая программа должна обеспечивать защиту от неверного ввода данных (иногда такую защиту называют «защитой от дурака» — fool proof).

Поскольку пользователь может вводить данные неверно сколько угодно раз, то надо использовать цикл с условием. С другой стороны, один раз обязательно надо ввести число, поэтому нужен цикл с постусловием. В отличие от предыдущей программы, теперь надо при каждом делении определять остаток (последняя цифра числа равна остатку от деления его на 10) и суммировать все остатки в специальной переменной.

#### Тест 1. Проверка ввода

```
Введите натуральное число:-21
Введите натуральное число:0
Введите натуральное число:91
Сумма цифр этого числа равна 10
```

#### Тест 2

```
Введите натуральное число:23
Сумма цифр этого числа равна 5
```

Текст программы.

```
#include <stdio.h>
#include <locale.h>
main()
{
    setlocale(LC_ALL,"Russian");
    int N, sum; // sum - сумма цифр числа
    sum = 0; // сначала сумму обнуляем
    do // начало цикла для проверки ввода N
    {
        printf ( "\nВведите натуральное число:" );
        scanf ( "%d", &N );
    }
    while ( N <= 0 ); // условие цикла «пока N <= 0»
    while ( N > 0 )
    {
        sum += N % 10;
        N /= 10;
    }
    printf ( "Сумма цифр этого числа равна %d\n", sum );
    return 0;
}
```



В следующей лекции:  
Продолжение операторы  
цикла