Информатика и ИКТ


# ПРОГРАММИРОВАНИЕ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ

## НАЧАЛА ПРОГРАММИРОВАНИЯ

9 класс

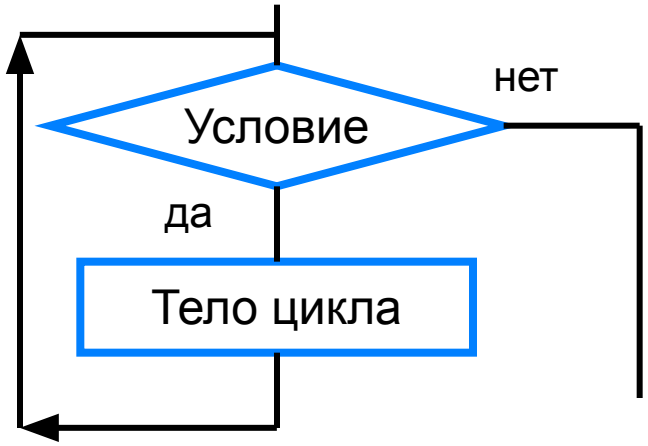
ИЗДАТЕЛЬСТВО БИНОМ

# Ключевые слова

- **while ( цикл-ПОКА)**

  **repeat (цикл-ДО)**
- **for (цикл с параметрами)**

# Программирование циклов с заданным условием продолжения работы



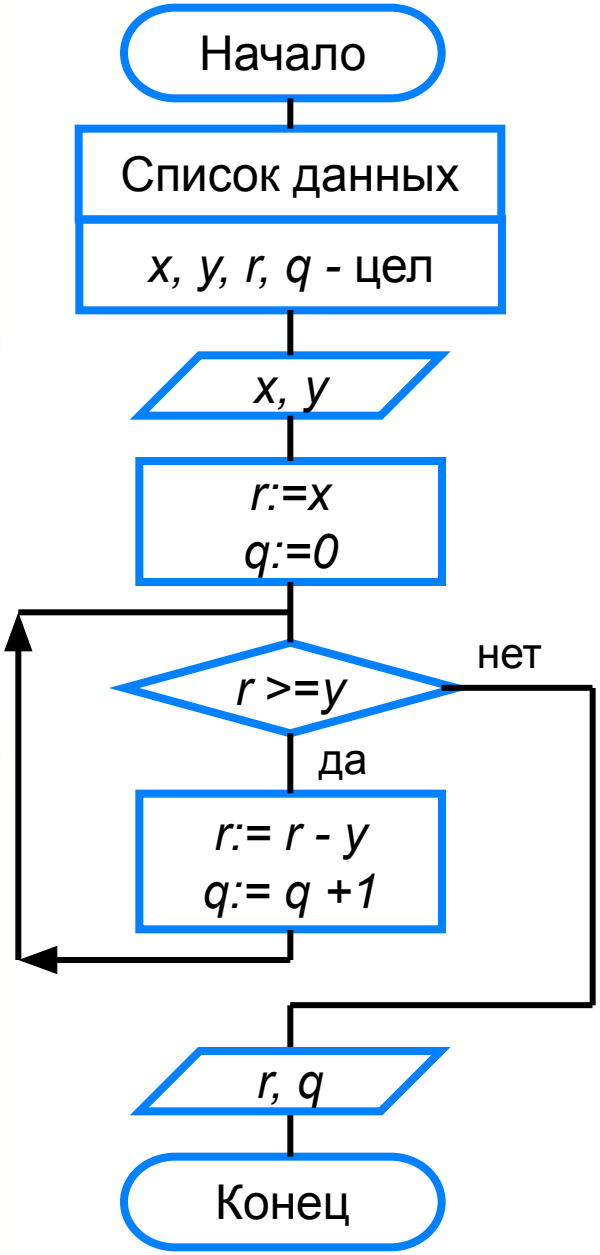

Общий вид оператора:

**while** <**условие**> **do** <**оператор**>

Здесь:

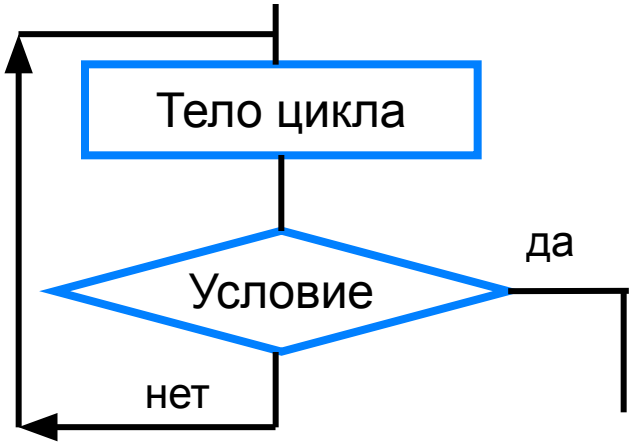
<**условие**> - логическое выражение; пока оно истинно, выполняется тело цикла;

<**оператор**> - простой или составной оператор, с помощью которого записано тело цикла.




```
program n_14;
  var x, y, q, r: integer;
begin
  writeln ('Частное и остаток');
  write ('Введите делимое x>>');
  readln (x);
  write ('Введите делитель y>>');
  read (y);
  r:=x;
  q:=0;
  while r>=y do
  begin
   r:=r-y;
   q:=q+1
  end;
  writeln ('Частное q=', q);
  writeln ('Остаток r=', r)
end.
```

# Программирование циклов с заданным условием окончания работы




Общий вид оператора:

**repeat** <**оператор1; оператор2; …;** > **until** <**условие**>

Здесь:

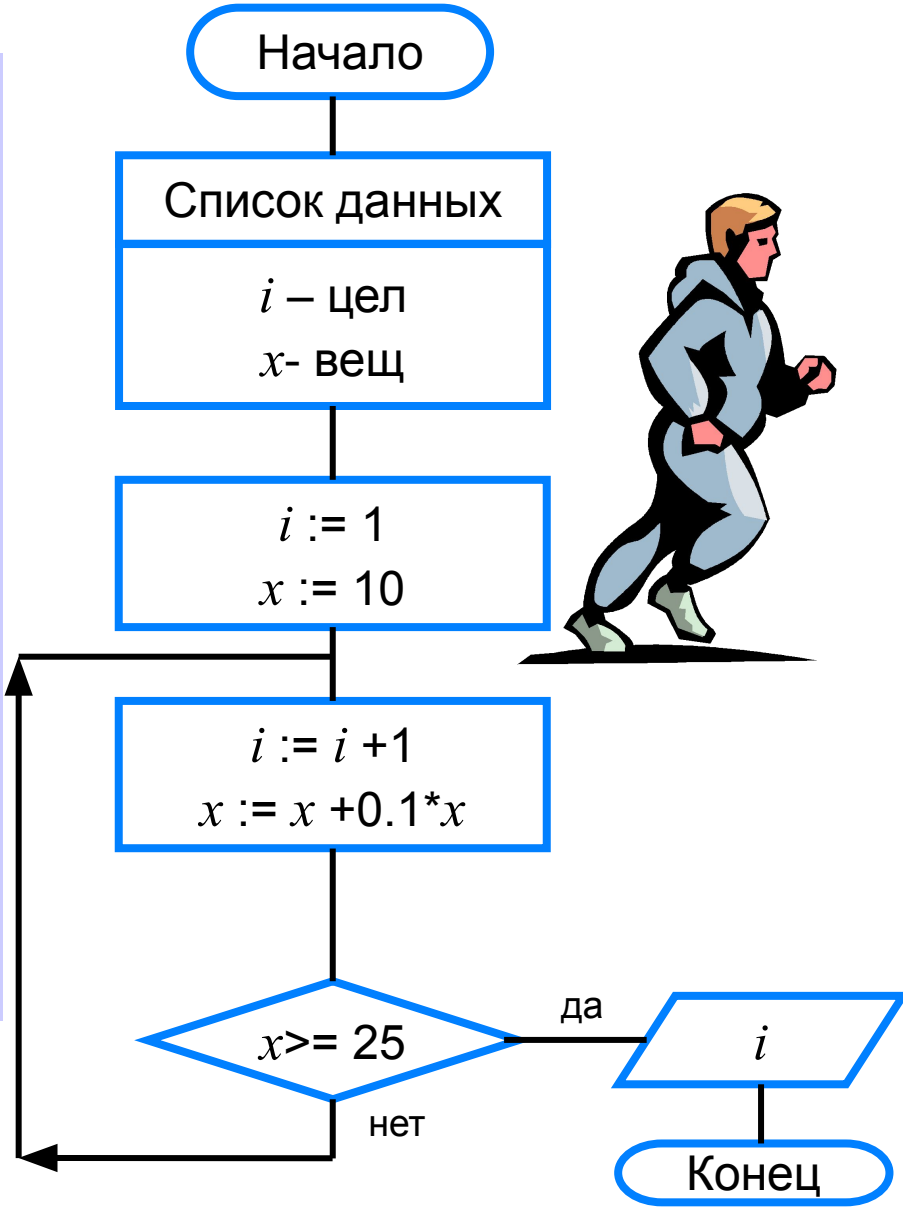
<**оператор1**>; <**оператор2**>; … - операторы, образующие тело цикла;

<**условие**> - логическое выражение; если оно ложно, то выполняется тело цикла.

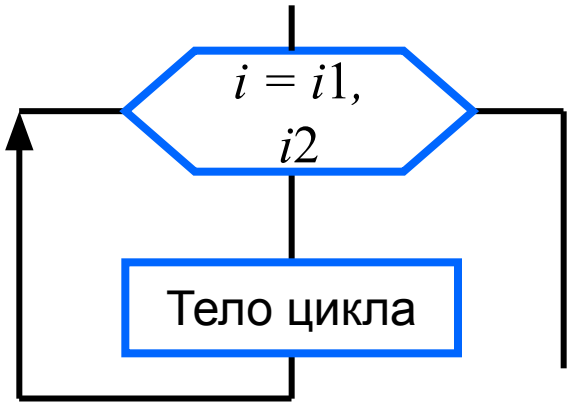
**Пример 17.** Спортсмен приступает к тренировкам по следующему графику: в первый день он должен пробежать 10 км; каждый следующий день следует увеличивать дистанцию на 10% от нормы предыдущего дня. Как только дневная норма достигнет или превысит 25 км, необходимо прекратить её увеличение и далее пробегать ежедневно ровно 25 км. Начиная с какого дня спортсмен будет пробегать 25 км?

Пусть $x$ — количество километров, которое спортсмен пробежит в некоторый $i$-й день. Тогда в следующий $(i + 1)$-й день он пробежит $x + 0{,}1x$ километров ($0{,}1x$ — это 10% от $x$).

```
program n_15;
  var i: integer; x: real;
begin
  writeln ('График тренировок');
  i:=1;
  x:=10;
  repeat
  i:=i+1;
  x:=x+0.1*x;
  until x>=25;
  writeln ('Начиная с ', i, '-го  дня спортсмен будет пробегать 25 км')
end.
```



# Программирование циклов с заданным числом повторений




Общий вид оператора:

**for** <**параметр**>:=<**начальное_значение**>

**to** <**конечное_значение**> **do** <**оператор**>

Здесь:

<**параметр**> - переменная целого типа;

<**начальное_значение**> и <**конечное_значение**> - выражения того же типа, что и параметр;

<**оператор**> - простой или составной оператор - тело цикла.

После каждого выполнения тела цикла происходит увеличение на единицу параметра цикла; условие выхода из цикла - превышение параметром конечного значения.

**Пример 19.** Составим алгоритм вычисления степени с натуральным показателем $n$ для любого вещественного числа $a$.

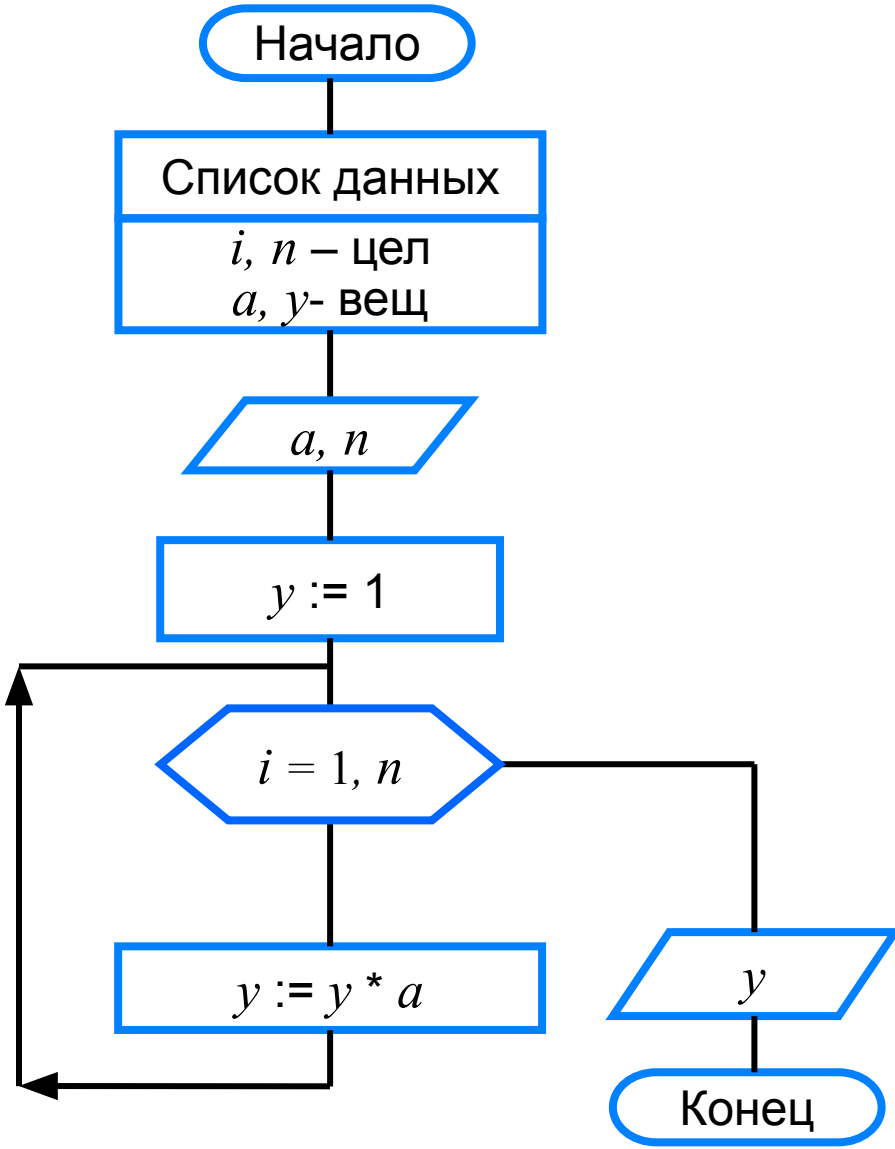
По определению:

$$a^1 = a,\ a^n = \underbrace{a \cdot a \cdot \ldots a}_{n \text{ сомножителей}},\ a \in R,\ n \in N,\ n \geq 2.$$

При составлении алгоритма воспользуемся единой формулой, в которой число умножений равно показателю степени:

$$a^n = \underbrace{1 \cdot a \cdot a \cdot \ldots a}_{n \text{ умножений}}$$

```
program n_16;
  var i,n:integer;a,y:real;
begin
  writeln ('Возведение в степень');
  write ('Введите основание a>>');
  readln (a);
  write ('Введите показатель n>>');
  readln (n);
  y:=1;
  for i:=1 to n do y:=y*a;
  writeln ('y=', y)
end.
```



Исполним этот алгоритм для $a = 4$ и $n = 3$.

| Шаги алгоритма | Операция | Переменные | | | | Условие |
|---|---|---|---|---|---|---|
| | | $a$ | $n$ | $y$ | $i$ | $i <= n$ |
| 1 | Ввод $a$, $n$ | 4 | 3 | – | – | |
| 2 | $y := 1$ | 4 | 3 | 1 | – | |
| 3 | $i := 1$ | 4 | 3 | 1 | 1 | |
| 4 | $i <= n$ | | | | | 1 <=3 (Да) |
| 5 | $y := y \cdot a$ | 4 | 3 | 4 | 1 | |
| 6 | $i := i + 1$ | 4 | 3 | 4 | 2 | |
| 7 | $i <= n$ | | | | | 2 <=3 (Да) |
| 8 | $y := y \cdot a$ | 4 | 3 | 16 | 2 | |
| 9 | $i := i + 1$ | 4 | 3 | 16 | 3 | |
| 10 | $i <= n$ | | | | | 3 <=3 (Да) |
| 11 | $y := y \cdot a$ | 4 | 3 | 64 | 3 | |
| 12 | $i := i + 1$ | 4 | 3 | 64 | 4 | |
| 13 | $i <= n$ | | | | | 4 <=3 (Да) |

# Различные варианты программирования циклического алгоритма

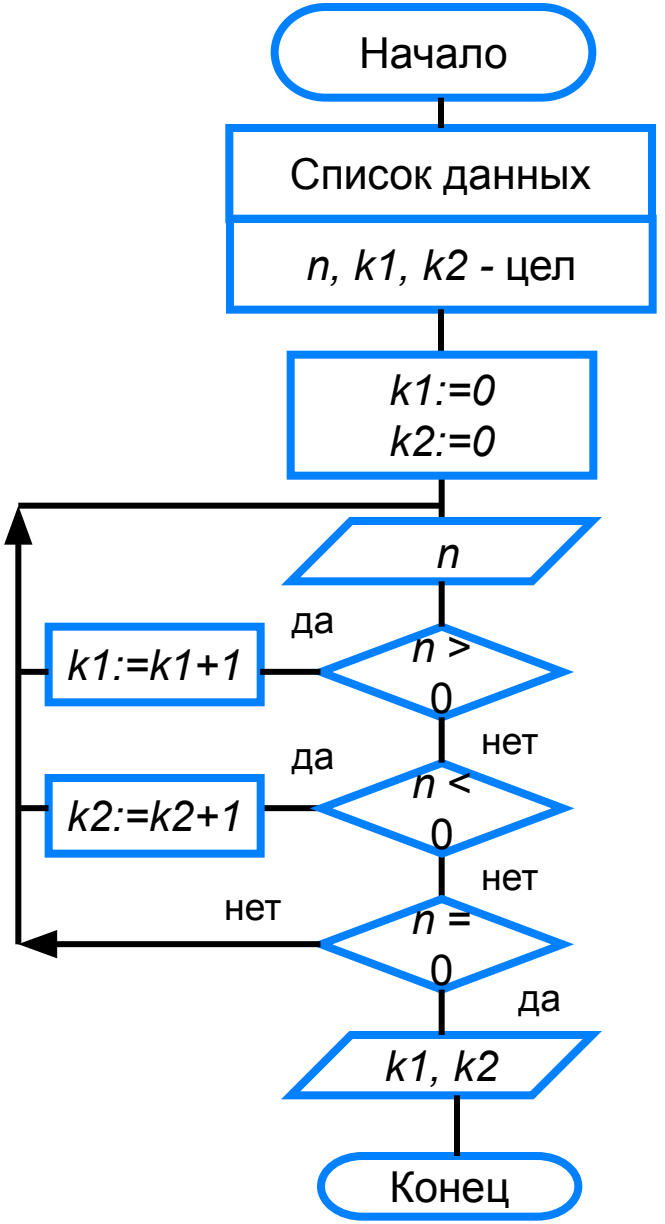
Для решения одной и той же задачи могут быть созданы разные программы.

Организуем ввод целых чисел и подсчёт количества введённых положительных и отрицательных чисел. Ввод должен осуществляться до тех пор, пока не будет введён ноль.

В задаче в явном виде задано условие окончания работы.

Воспользуемся оператором **repeat**.

```
program n_17;
  var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  repeat
    write ('Введите целое число>>');
    readln (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  until n=0;
  writeln ('Введено:');
  writeln ('положительных чисел – ', k1);
  writeln ('отрицательных чисел – ', k2)
end.
```

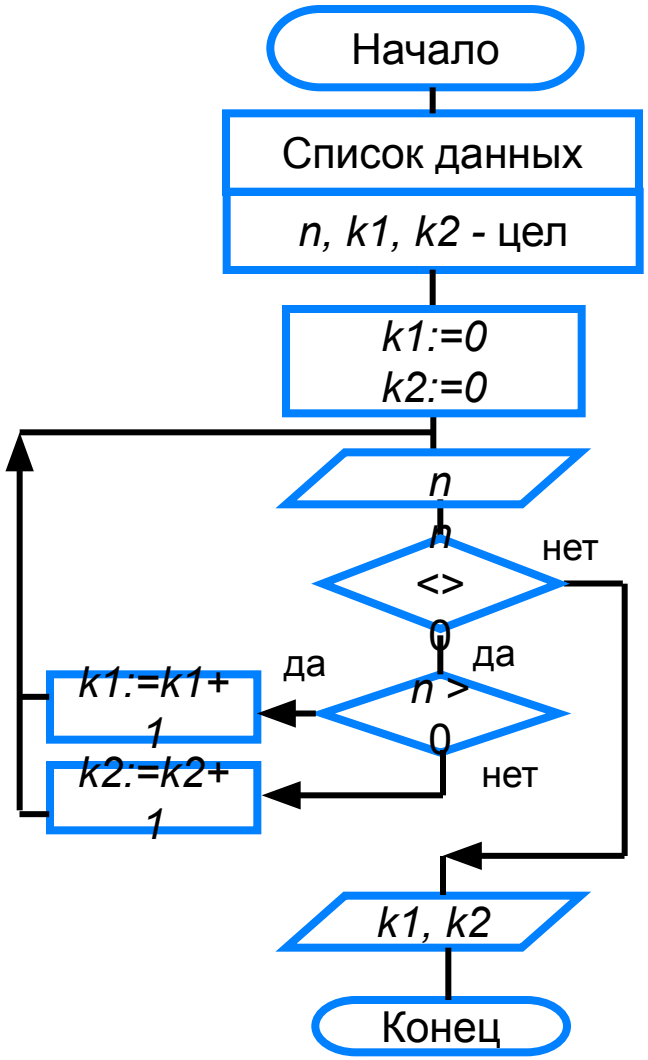


Ввод осуществляется до тех пор, пока не будет введён ноль.

Работа продолжается, пока $n \neq 0$.

Воспользуемся оператором **while**:

```
program n_18;
  var n, k1, k2: integer;
begin
  k1:=0;
  k2:=0;
  while n<>0 do
  begin
    writeln ('Введите целое число>>');
    read (n);
    if n>0 then k1:=k1+1;
    if n<0 then k2:=k2+1;
  end;
  writeln ('Введено:');
  writeln ('положительных – ', k1);
  writeln ('отрицательных – ', k2)
end.
```



# Самое главное

В языке Паскаль имеются три вида операторов цикла:

***while*** цикл-ПОКА)

***repeat*** (цикл-ДО)

***for (***цикл с параметром).

Если число повторений тела цикла известно, то лучше воспользоваться оператором ***for***; в остальных случаях используются операторы ***while*** и ***repeat***.

# Вопросы и задания

Какой из трёх рассмотренных операторов цикла является, по вашему мнению, основным, т. е. таким, что им можно заменить два других оператора цикла?

Сколько раз будет повторён цикл? Какими будут значения переменных после выполнения этой последовательности операторов?

| Пример входных данных | Пример выходных данных |
|---|---|
| Введите n> 5 | 5 x 1 = 5<br>5 x 2 = 10<br>5 x 3 = 15<br>5 x 4 = 20<br>5 x 5 = 25<br>5 x 6 = 30<br>5 x 7 = 35<br>5 x 8 = 40<br>5 x 9 = 45<br>5 x 10 = 50 |
| Введите n> 6 | 6! = 720 |

# Опорный конспект

В языке Паскаль имеются три вида операторов цикла: