Информатика

ПРОГРАММИРОВАНИЕ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ

НАЧАЛА ПРОГРАММИРОВАНИЯ

8 класс

ИЗДАТЕЛЬСТВО БИНОМ

Ключевые слова

- **while (цикл-ПОКА)**
- **for (цикл с параметрами)**

Программирование циклов с заданным условием продолжения работы

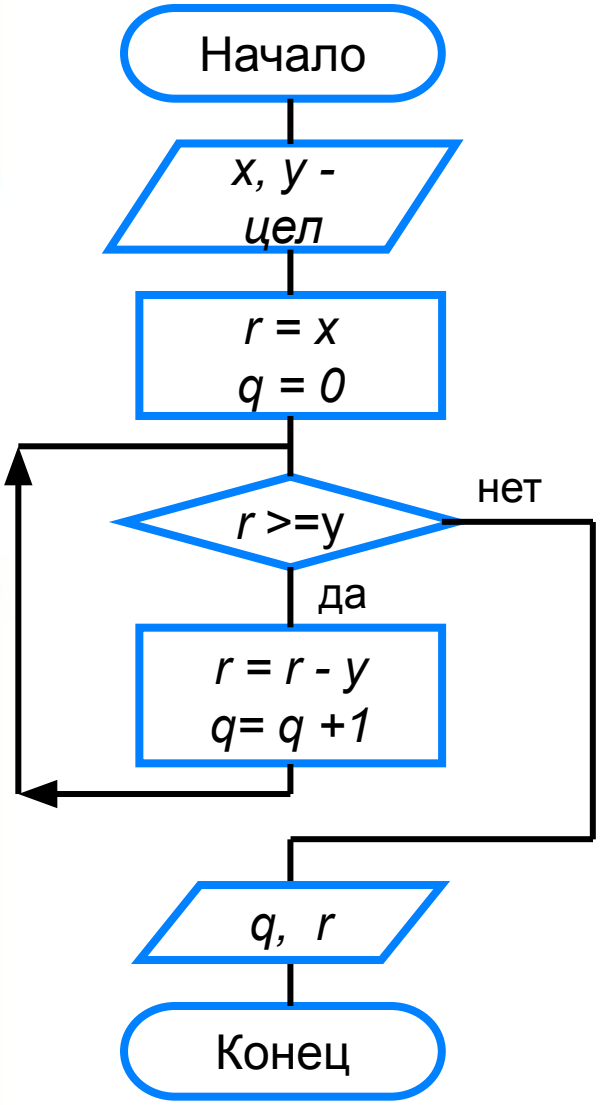
Общий вид оператора:

while <**условие**> :

<**оператор**>

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

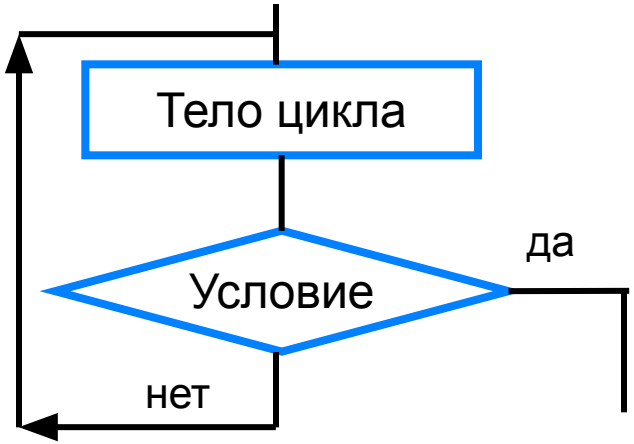
<**оператор**> - простой или составной оператор, с помощью которого записано тело цикла.




```
print ('Частное и остаток')
x = int(input (' Введите делимое x>>'))
y = int(input (' Введите делитель y>>'))
r = x
q = 0
while r >= y:
    r = r – y
    q += 1
print ('Частное  q=‘, q)
print ('Остаток  r=', r)
```

Программирование циклов с заданным условием окончания работы

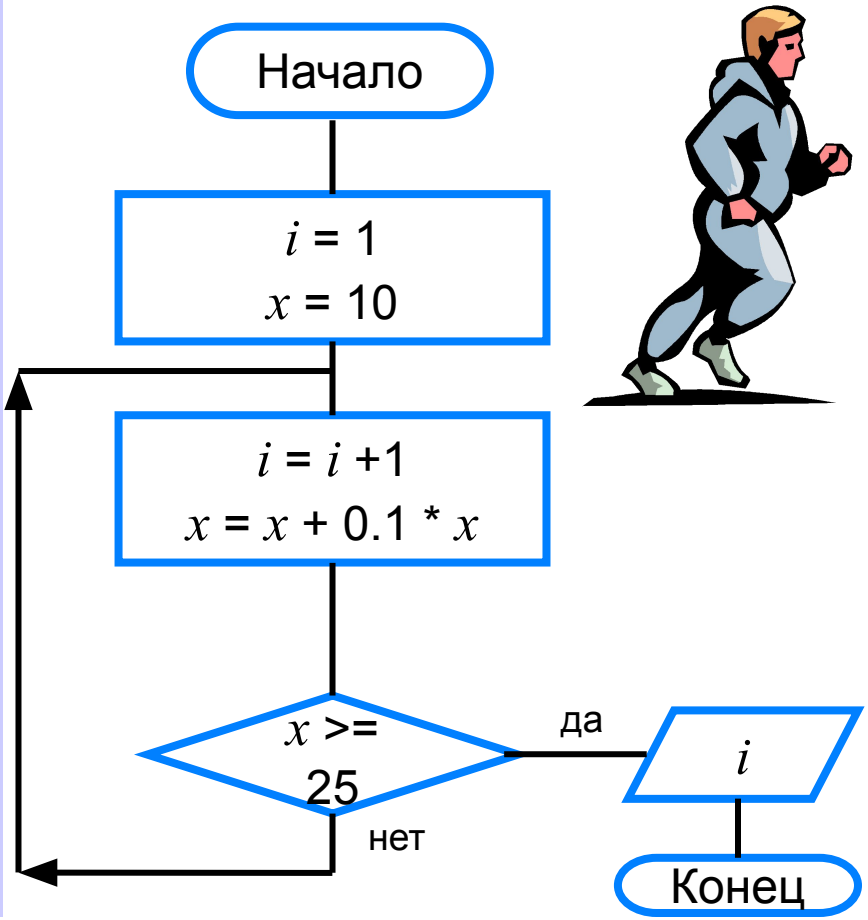
В языке Python нет цикла с заданным условием окончания работы, но его можно организовать с помощью цикла ***while****:*




```
while  True:
        <операторы>
        if <условие>: break
```

Такой цикл будет выполняться бесконечно, потому что условие ***True*** *всегда истинно. Оператора* ***break*** *обеспечивает выход из цикла если условие истинно (в переводе с англ. – «прервать», досрочный выход из цикла).*

```
print ('График тренировок')
i = 1
x = 10
while True:
        i += 1
        x = x + 0.1 * x
        if  x >= 25: break
print ('Начиная с ', i, '-го дня
спортсмен будет пробегать
25 км')
```

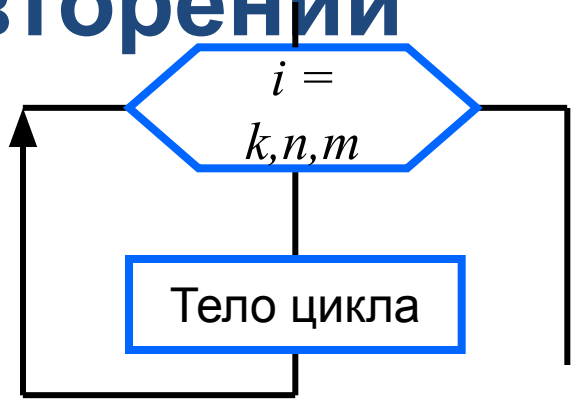


Программирование циклов с заданным числом повторений

Общий вид оператора:

for <параметр> **in range (k, n, m):**

<оператор>



<параметр> - переменная целого типа

range() – диапазон значений:

k – **начальное значение переменной** *(по умолчанию равен 0)*

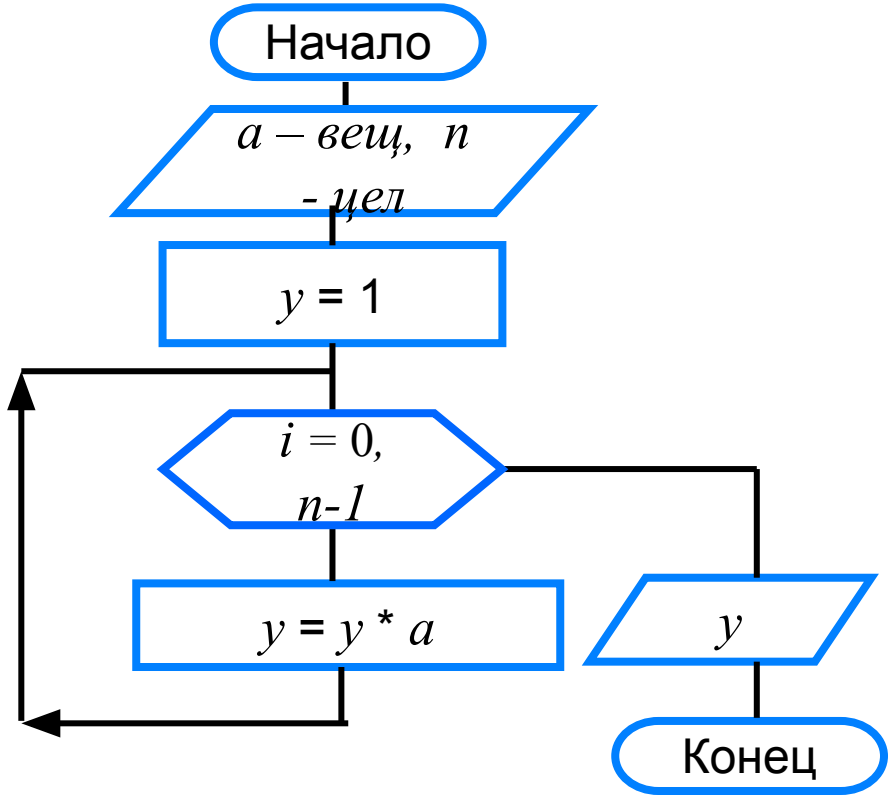
n –**конечное значение переменной,** *не включая последнее*

m – **шаг изменения переменной,** *по умолчанию равен 1*

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

```
# Возведение в степень
a= float(input (' Введите основание a>>'))
n= int(input (' Введите показатель n>>'))
y=1
for i  in  range(n):
      y = y * a
print('y=', y)
```

цикл будет работать от 0 до n-1
ровно n раз



Различные варианты программирования циклического алгоритма

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

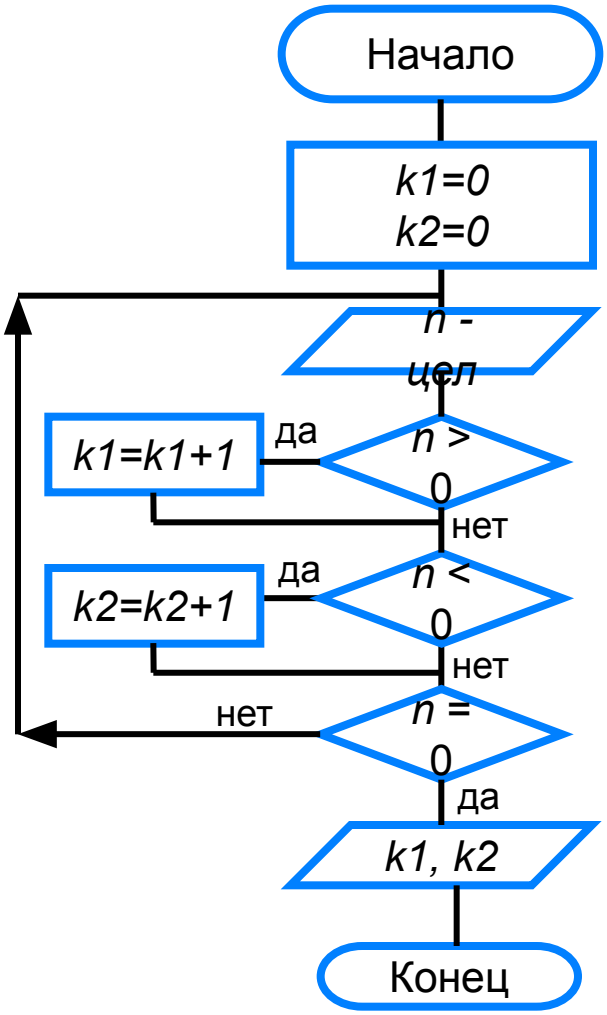
Организуем **ввод целых чисел и подсчёт количества введённых положительных и отрицательных чисел.**

Ввод должен осуществляться до тех пор, пока не будет введён ноль.

В задаче в явном виде задано условие окончания работы.

Воспользуемся циклом с постусловием

```
k1 = k2 = 0
while True:
    n = int (input (' Введите целое
            число>>'))
    if n > 0:
        k1 += 1
    if n < 0:
        k2 += 1
    if n == 0: break
print ('Введено:')
print('положительных чисел -', k1)
print('отрицательных чисел -', k2)
```

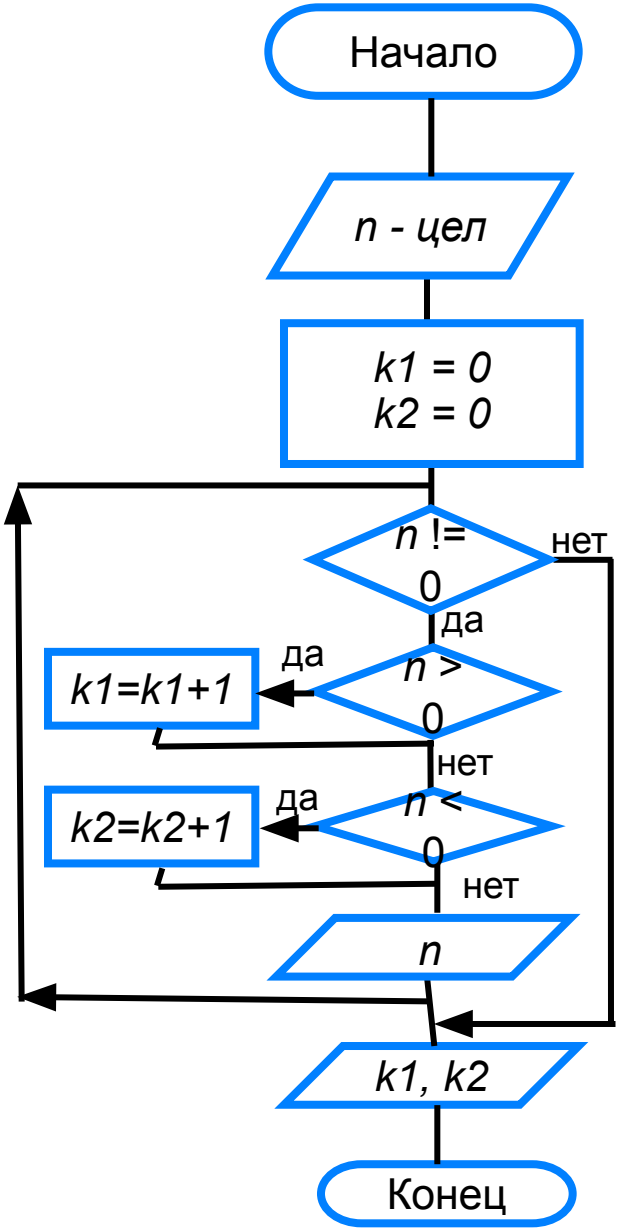


Ввод осуществляется до тех пор, пока не будет введён ноль.

Работа продолжается, пока $n \neq 0$.

Воспользуемся оператором **while**:

```
n = int (input (' Введите целое число>>'))
k1=k2 = 0
while n != 0:
        if n > 0:
                k1 += 1
        if n < 0:
                k2 += 1
        n = int (input (' Введите целое число>>'))
print ('Введено:')
print ('положительных - ', k1);
print ('отрицательных - ', k2)
```



Самое главное

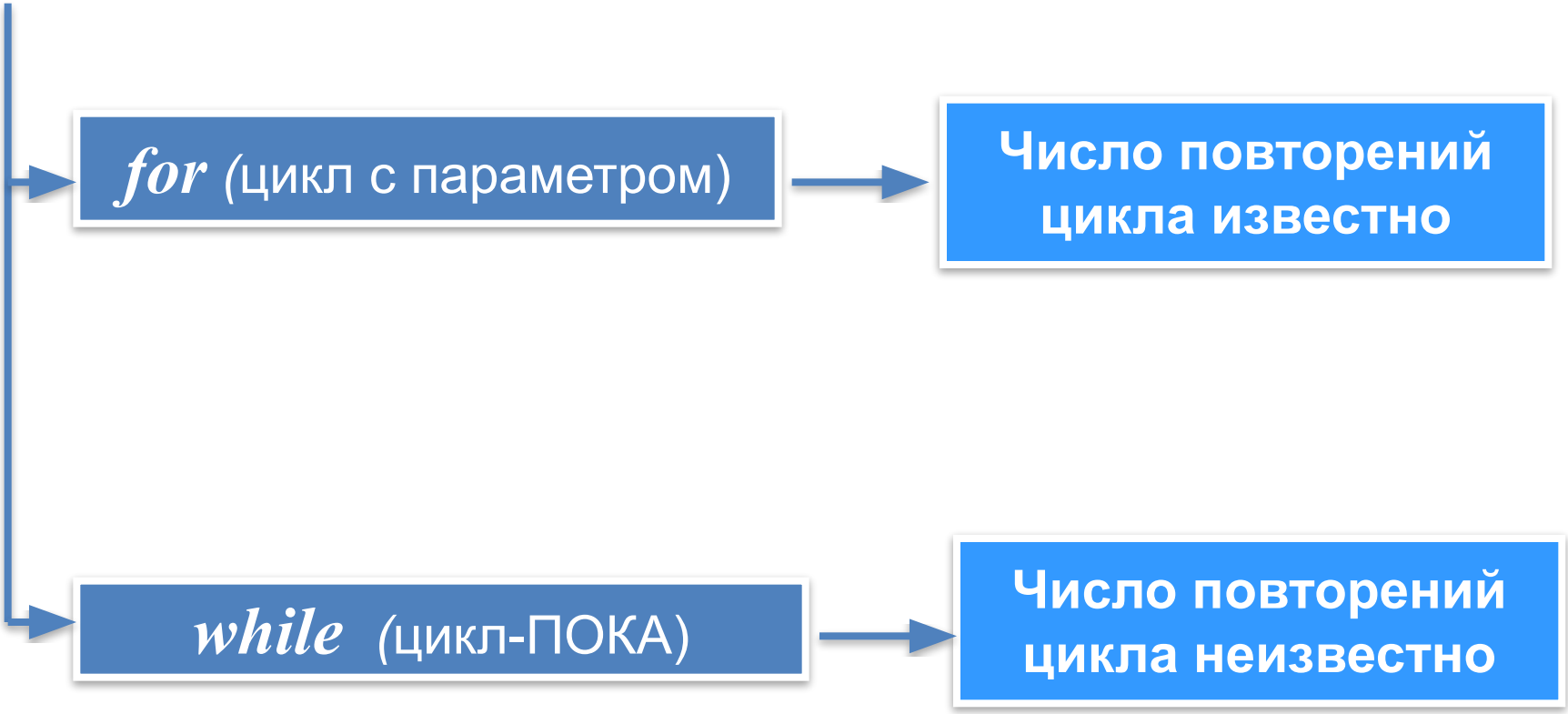
В языке Python имеются два вида операторов цикла:

while (цикл-ПОКА)

for (цикл с параметром).

Если число повторений тела цикла известно, то лучше воспользоваться оператором ***for***;

в остальных случаях используется оператор ***while***

Опорный конспект

В языке Python имеются два вида операторов цикла: