Структурные операторы языка Pascal. Операторы цикла.

Циклы

Цикл – это многократное выполнение одинаковой последовательности действий.

- цикл с известным числом шагов (цикл с переменной);
- цикл с неизвестным числом шагов (цикл с условием).

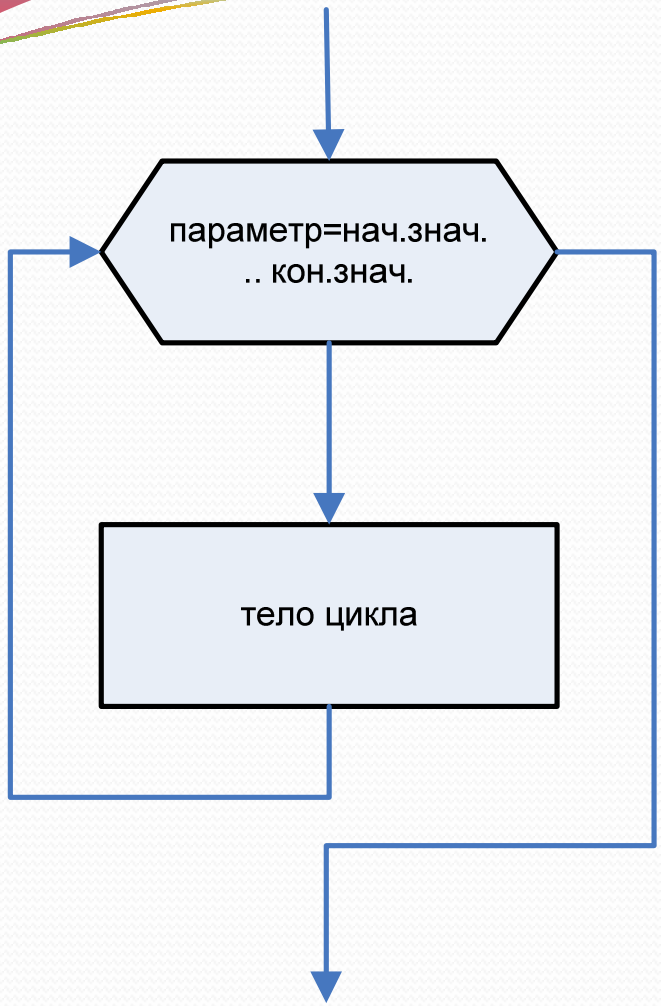
Цикл с переменной (параметром) – цикл FOR.

Увеличение переменной на 1:

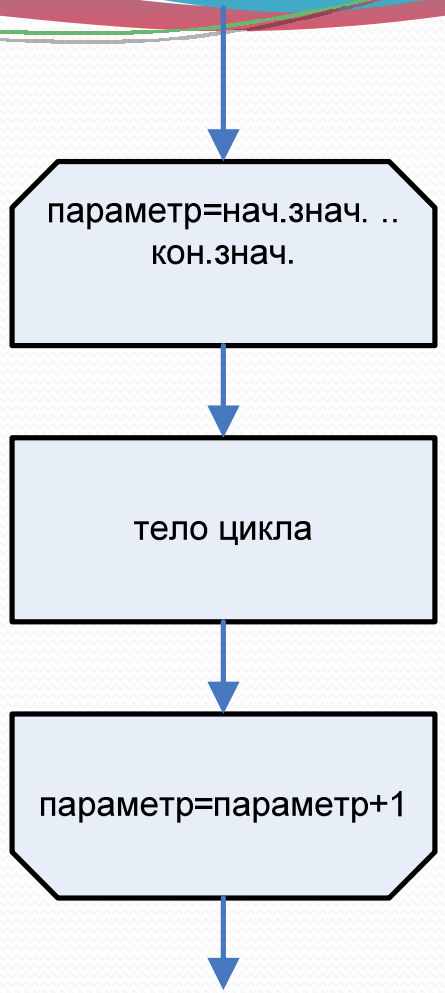
```
for <переменная> := <начальное значение> to
    <конечное значение> do begin
  {тело цикла}
end;
```

Уменьшение переменной на 1:

```
for <переменная> := <начальное значение>
          downto
    <конечное значение> do begin
  {тело цикла}
end;
```



а



б

Тело цикла повторяется для каждого значения параметра от начального до конечного.

Особенности:

- переменная цикла может быть только целой (integer)
- шаг изменения переменной цикла всегда равен 1 (to) или -1 (downto)
- если в теле цикла только один оператор, слова **begin** и **end** можно не писать:

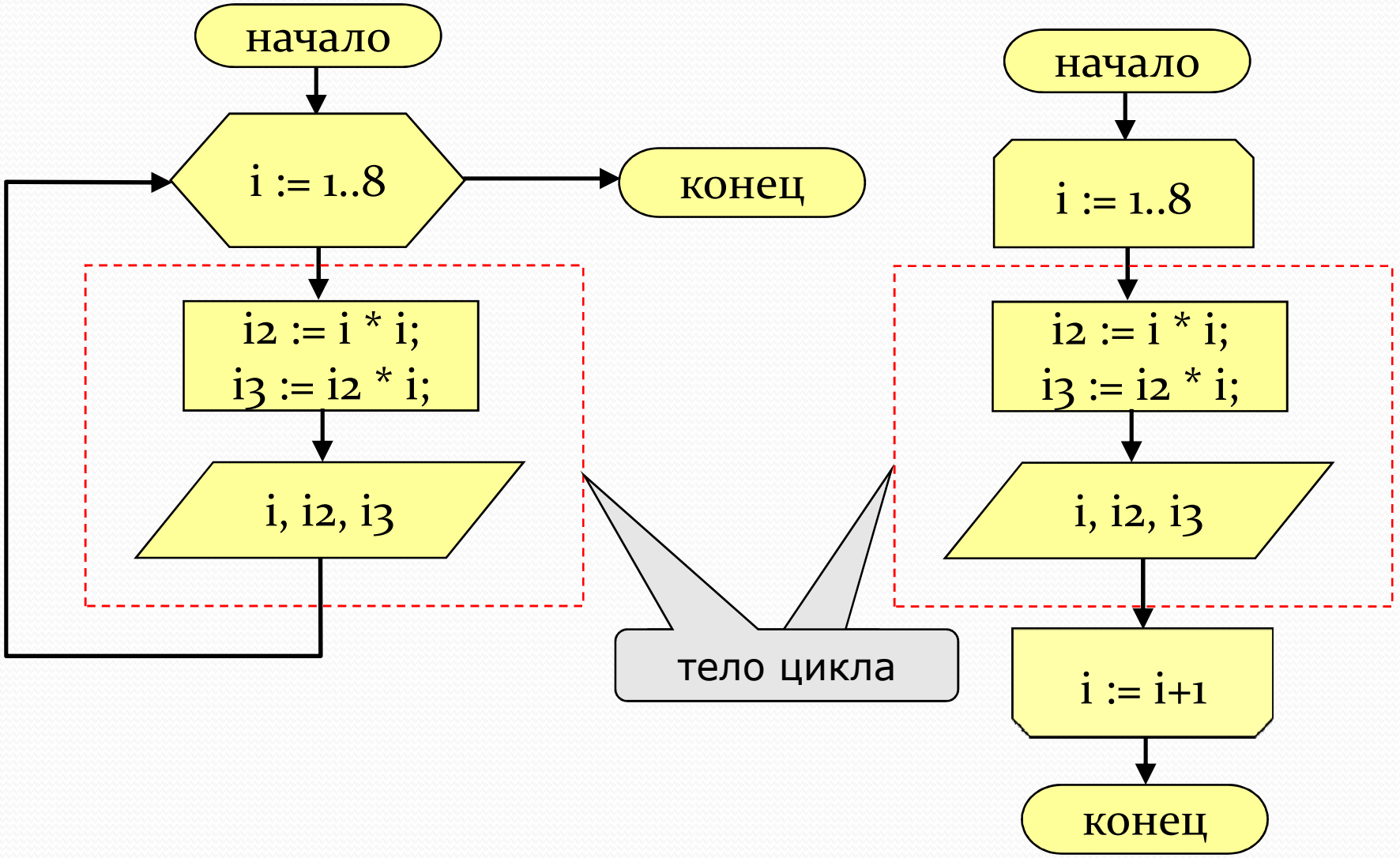
```
for i:=1 to 8 do
  writeln('Привет');
```

- если конечное значение меньше начального, цикл (to) не выполняется ни разу и наоборот с downto.

Пример №1: Вывести на экран квадраты и кубы целых чисел от 1 до 8.

Особенность: одинаковые действия выполняются 8 раз.

Алгоритм (с блоком "цикл")



Программа

```
program Example_1;
var i, i2, i3: integer;
begin
  for i:=1 to 8 do begin
    i2 := i*i;
    i3 := i2*i;
    writeln(i, i2, i3);
  end;
readln;
end.
```

переменная цикла

начальное значение

конечное значение

Цикл с уменьшением переменной

Задача: Вывести на экран квадраты и кубы целых чисел от 8 до 1 (в обратном порядке).

Особенность: переменная цикла должна уменьшаться.

Решение:

```
...
for i:=8 downto 1 do begin
  i2 := i*i;
  i3 := i2*i;
  writeln(i, i2, i3);
end;
...
```

Сколько раз выполняется цикл?

```
a := 1;
for i:=1 to 3 do a := a+1;
```

a = 4

```
a := 1;
for i:=3 to 1 do a := a+1;
```

a = 1

```
a := 1;
for i:=1 downto 3 do a := a+1;
```

a = 1

```
a := 1;
for i:=3 downto 1 do a := a+1;
```

a = 4

Как изменить шаг?

Задача: Вывести на экран квадраты и кубы нечётных целых чисел от 1 до 9.

Особенность: переменная цикла должна увеличиваться на 2.

Проблема: в Паскале шаг может быть 1 или -1.

Решение:

```
...
for i:=1 to 9 do begin
  if i mod 2 = 1 then begin
    i2 := i*i;
    i3 := i2*i;
    writeln(i, i2, i3);
  end;
end;
...
```

выполняется только для нечетных i

Как изменить шаг? – II

Идея: Надо вывести всего 5 чисел, переменная k изменяется от 1 до 5. Начальное значение i равно 1, с каждым шагом цикла i увеличивается на 2.

Решение:

```
...
i := 1;
for k:=1 to 5 do begin
  i2 := i*i;
  i3 := i2*i;
  writeln(i, i2, i3);
  i := i + 2;
end;
...
```

Как изменить шаг? – III

Идея: Надо вывести всего 5 чисел, переменная k изменяется от 1 до 5. Зная k, надо рассчитать i.

k	1	2	3	4	5
i	1	3	5	7	9

i = 2k-1

Решение:

```
...
for k:=1 to 5 do begin
  i := 2*k - 1;
  i2 := i*i;
  i3 := i2*i;
  writeln(i, i2, i3);
end;
...
```

Цикл с предусловием – цикл While

Оператор **while** используется в том случае, когда некоторую последовательность действий надо выполнить несколько раз, причем необходимое число повторений заранее неизвестно и может быть определено только во время работы программы, то есть в процессе вычислений.

Цикл с предусловием – цикл While

```
while <условие> do begin
   {тело цикла}
end;
```

Особенности:

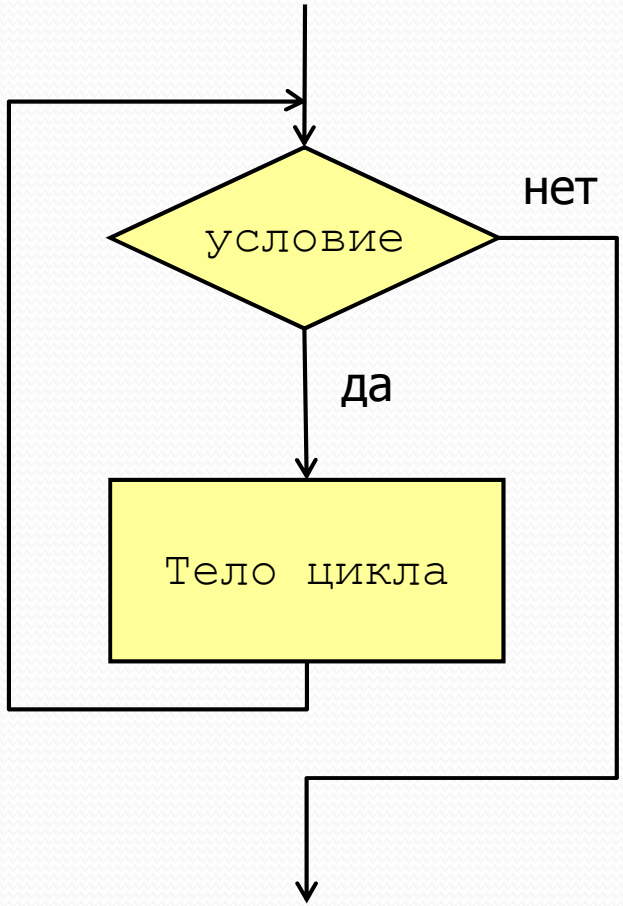
- можно использовать сложные условия:

```
while (a<b) and (b<c) do begin
   {тело цикла}
end;
```

- если в теле цикла только один оператор, слова **begin** и **end** можно не писать:

```
while a < b do
    a := a + 1;
```

Блок схема цикла с предусловием



Пока условие истинно повторяются операторы в теле цикла.

Цикл с предусловием

Особенности:

- условие пересчитывается каждый раз при входе в цикл
- если условие на входе в цикл ложно, цикл не выполняется ни разу

```
a := 4; b := 6;
while a > b do
   a := a – b;
```

- если условие никогда не станет ложным, программа зацикливается

```
a := 4; b := 6;
while a < b do
   d := a + b;
```

Пример №2: Ввести целое число и определить число цифр в нем.

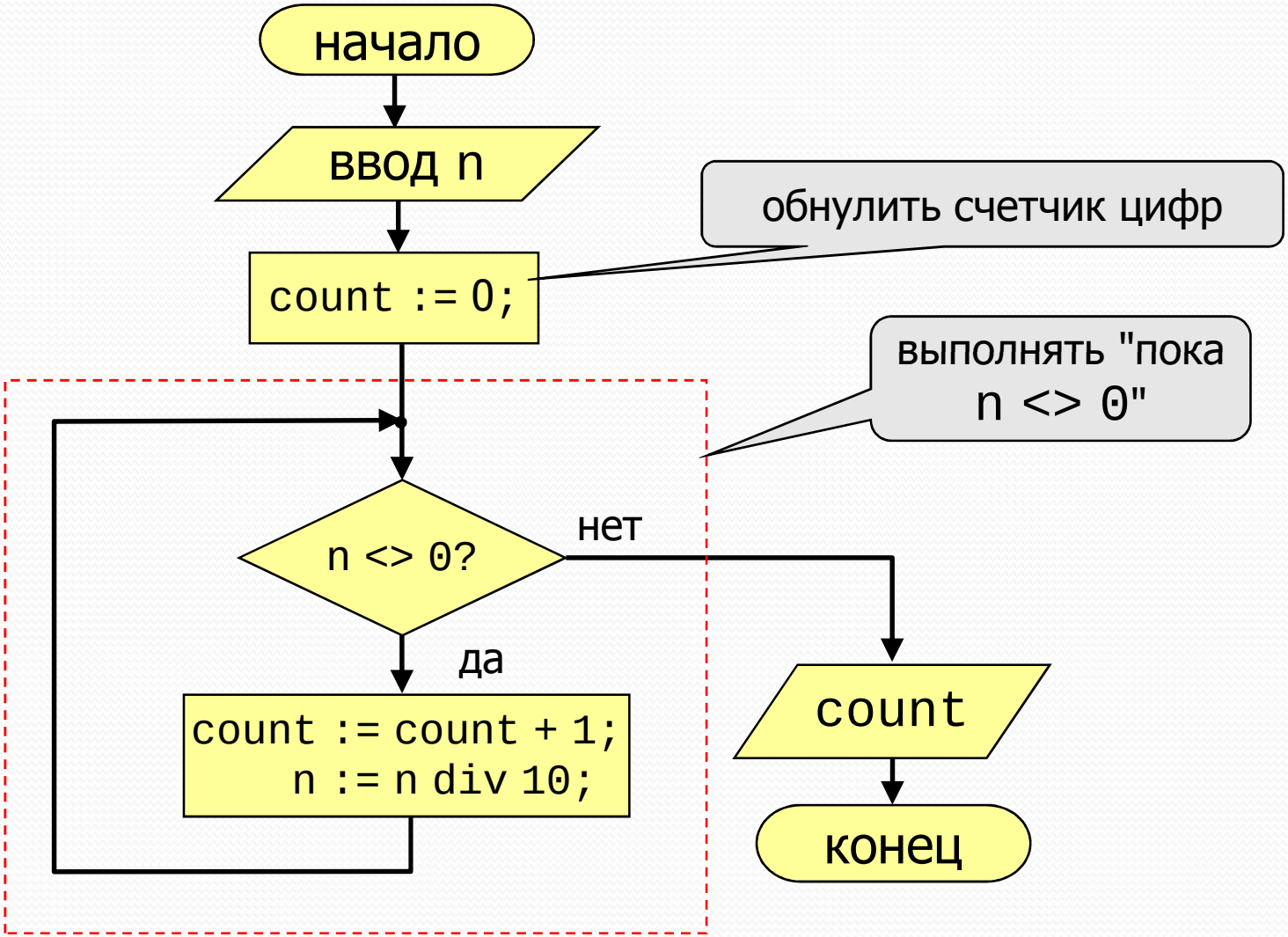
Идея решения: Отсекаем последовательно последнюю цифру, увеличиваем счетчик.

n	count
123	0
12	1
1	2
0	3

Проблема: Неизвестно, сколько шагов надо сделать.

Решение: Надо остановиться, когда n = 0, т.е. надо делать "пока n <> 0“.

Алгоритм


начало
ввод n
count := 0;
обнулить счетчик цифр
выполнять "пока n <> 0"
n <> 0?
нет
да
count := count + 1;
n := n div 10;
count
конец

Программа

```
program Example_2;
var n,n1,count: integer;
begin
  writeln('Введите целое число');
  readln(n); n1:=n;
  count:= 0;
  while n <> 0 do begin
    count:= count + 1;
    n:= n div 10;
  end;
  writeln('В числе ', n1, ' нашли ',
           count, ' цифр');
readln
end.
```

выполнять "пока n <> 0"

Сколько раз выполняется цикл?

```
a := 4; b := 6;
while a < b do a := a + 1;
```

2 раза
a = 6

```
a := 4; b := 6;
while a < b do a := a + b;
```

1 раз
a = 10

```
a := 4; b := 6;
while a > b do a := a + 1;
```

0 раз
a = 4

```
a := 4; b := 6;
while a < b do b := a - b;
```

1 раз
b = -2

```
a := 4; b := 6;
while a < b do a := a - 1;
```

зацикливание

Замена for на while и наоборот

```
for i:=a to b do begin
  {тело цикла}
end;
```

```
i := a;
while i <= b do begin
  {тело цикла}
  i := i + 1;
end;
```

```
for i:=a downto b do
  begin
  {тело цикла}
end;
```

```
i := a;
while i >= b do begin
  {тело цикла}
  i := i - 1;
end;
```

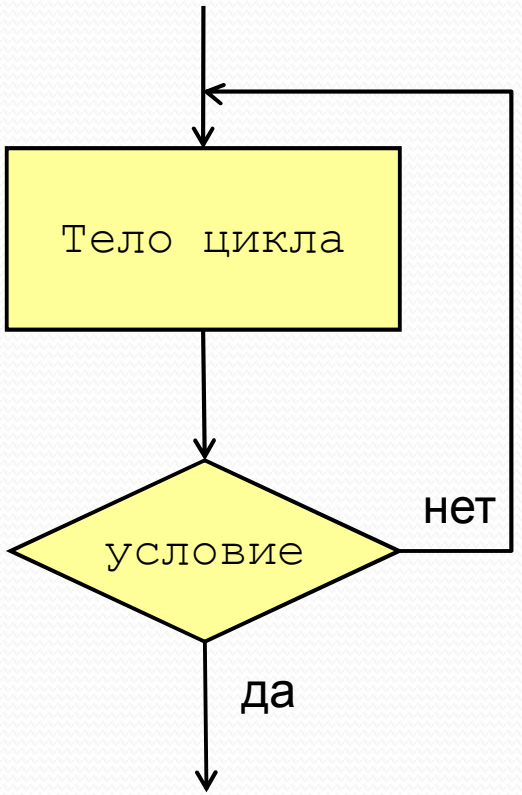
Замена цикла for на while возможна всегда.

Замена while на for возможна только тогда, когда можно заранее рассчитать число шагов цикла.

Цикл с постусловием – цикл Repeat

Цикл с постусловием – это цикл, в котором проверка условия выполняется в конце цикла.

```
repeat
{тело цикла};
until <условие> ;
```



Операторы в теле цикла будут повторяться до тех пор, пока условие не станет истинным.

Тело цикла выполнится хотя бы один раз.

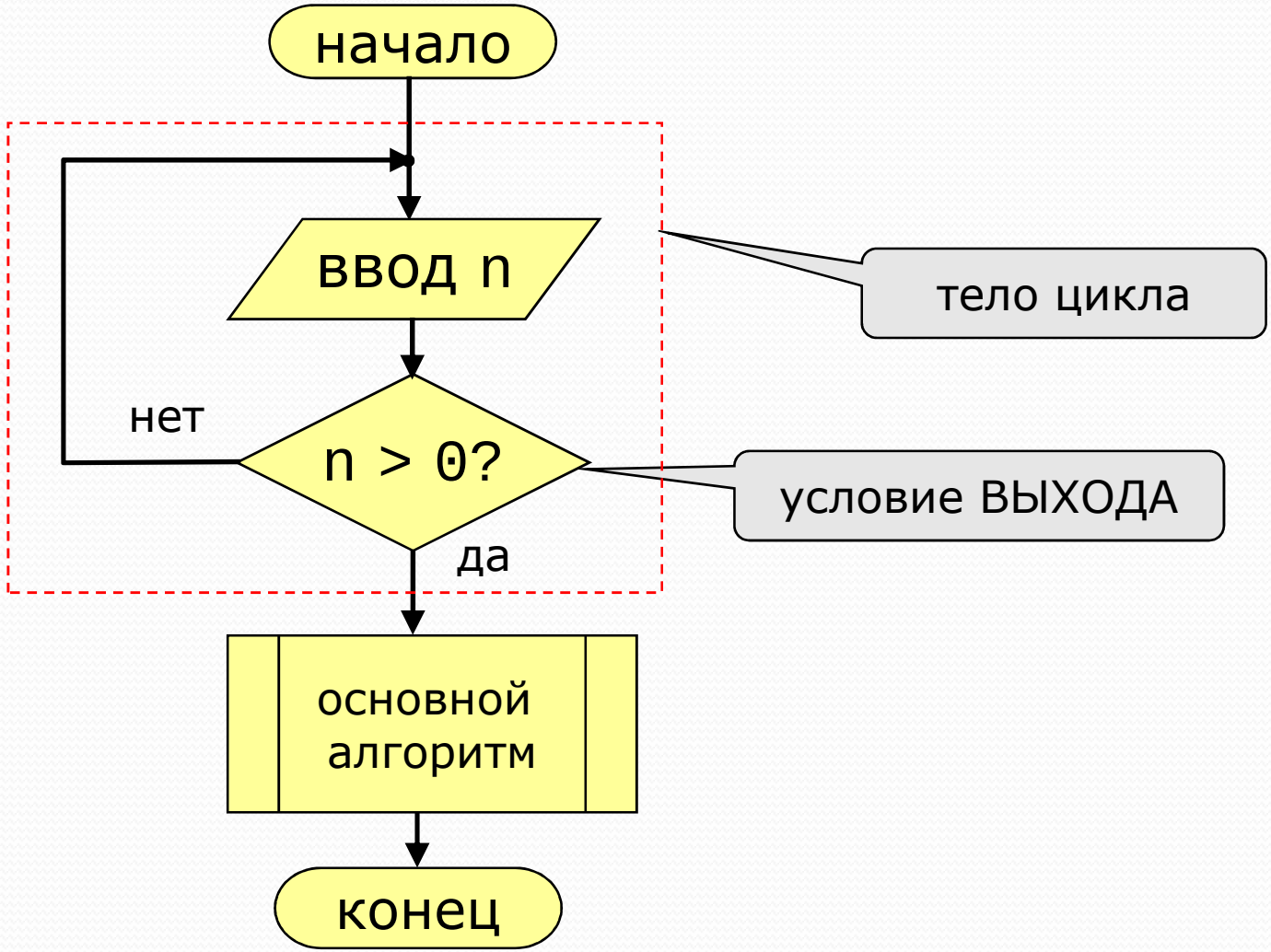
Задача: Ввести целое положительное число и определить число цифр в нем.

Проблема: Как не дать ввести отрицательное число или ноль?

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

Особенность: Один раз тело цикла надо сделать в любом случае, следовательно проверку условия цикла надо делать в конце цикла.

Цикл с постусловием: алгоритм



Программа

```
program Example_2;
var n,n1,count: integer;
begin
  repeat
    writeln('Введите положительное число');
    readln(n);
  until n > 0;
  ... { основной алгоритм }
end.
```

условие ВЫХОДА

Особенности:

- тело цикла всегда выполняется хотя бы один раз
- после слова until ("до тех пор, пока не…") ставится условие ВЫХОДА из цикла

Сколько раз выполняется цикл?

```
a := 4; b := 6;
repeat a := a + 1; until a > b;
```

3 раза
a = 7

```
a := 4; b := 6;
repeat a := a + b; until a > b;
```

1 раз
a = 10

```
a := 4; b := 6;
repeat a := a + b; until a < b;
```

зацикливание

```
a := 4; b := 6;
repeat b := a - b; until a < b;
```

2 раза
b = 6

```
a := 4; b := 6;
repeat a := a + 2; until a < b;
```

зацикливание

Стандартные процедуры для работы с циклом

Break – реализует немедленный выход из цикла.

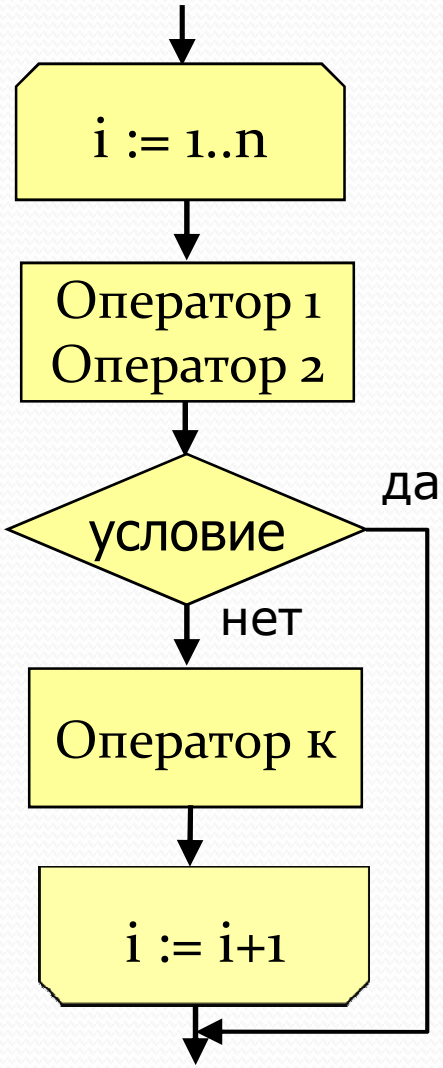
Особенность: действие процедуры заключается в передаче управления оператору, стоящему сразу после оператора цикла.

Continue – обеспечивает досрочное завершение очередного прохода цикла.

Особенность: действие процедуры заключается в передаче управления в самый конец циклического оператора.

Процедура **Break**

```
for i:=1 to n do
begin
      оператор 1;
      оператор 2;
      if условие then break;
      оператор k-1;
      оператор k;
end;
```



Процедура **Continue**

```
for i:=1 to n do
begin
    оператор 1;
    оператор 2;
    if условие then continue;
    оператор k-1;
    оператор k;
end;
```