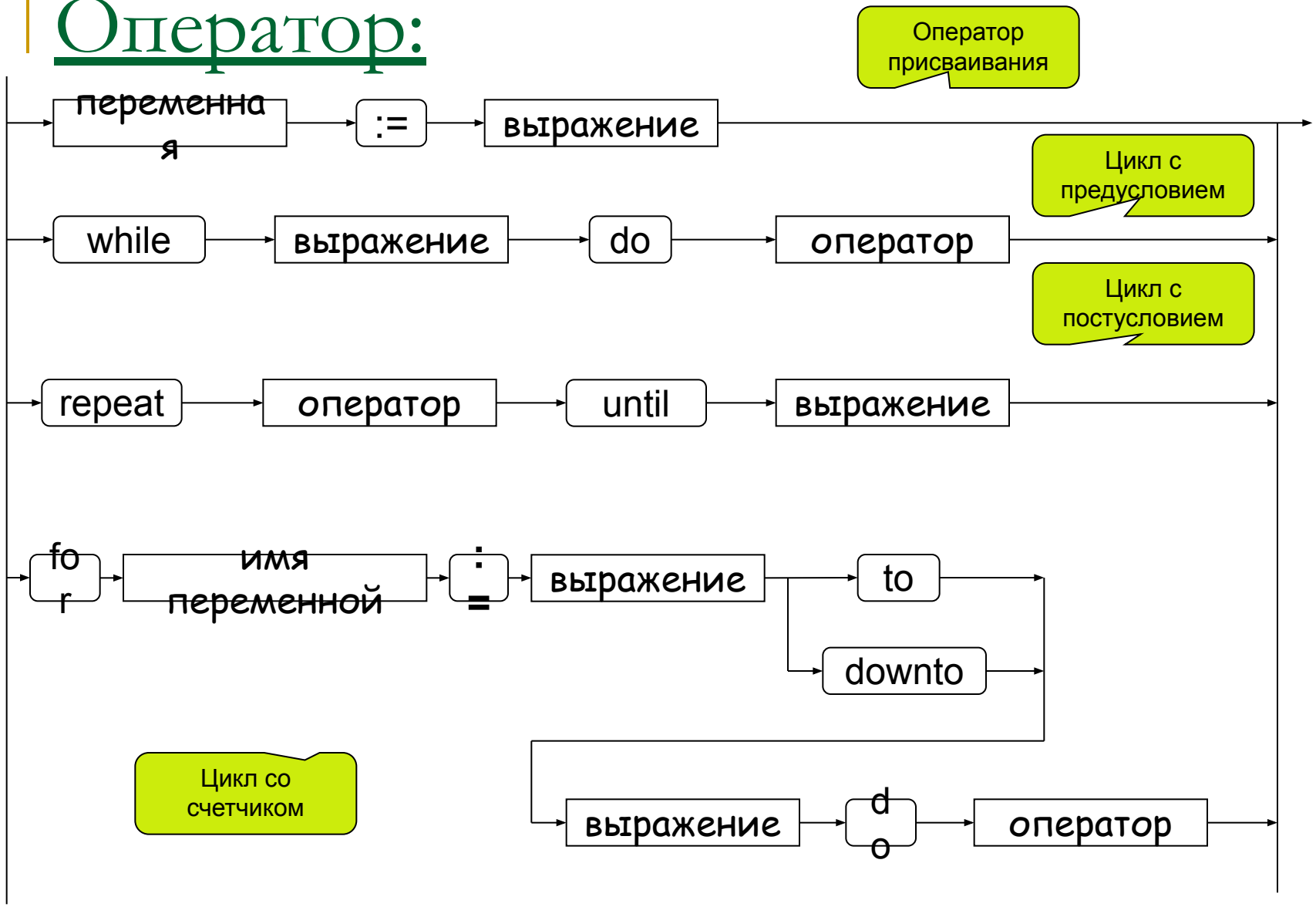
# Программирование на языке Паскаль

## Тема 5. Оператор цикла

# Оператор:


Оператор присваивания
переменная
:=
выражение
Цикл с предусловием
while
выражение
do
оператор
Цикл с постусловием
repeat
оператор
until
выражение
for
имя переменной
:=
выражение
to
downto
Цикл со счетчиком
выражение
do
оператор

# Цикл со счетчиком

## 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;
```

# Цикл со счетчиком

**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 (проверка условия в начале цикла, цикл с предусловием)

# Цикл со счетчиком

**Особенности:**

- в теле цикла не разрешается изменять переменную цикла (почему?)
- при изменении начального и конечного значения внутри цикла количество шагов не изменится:

```
n := 8;
for i:=1 to n do begin
  writeln('Привет');
  n := n + 1;
end;
```

нет зацикливания

# Цикл с переменной

**Особенности:**

- после выполнения цикла **во многих системах** устанавливается первое значение переменной цикла, при котором нарушено условие:

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

i=9

НЕ ДОКУМЕНТИРОВАНО

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

i=0

# 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.

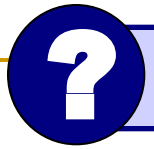
**Особенность:** переменная цикла должна увеличиваться на 2.

**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:4, i2:4, i3:4);
  end;
end;
...
```

выполняется только для нечетных **i**



**Что плохо?**

# 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:4, i2:4, i3:4);
  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:4, i2:4, i3:4);
end;
...
```

# Задания

**"4": Ввести a и b и вывести квадраты и кубы чисел от a до b.**

**Пример:**

```
Введите границы интервала:
4 6
  4   16    64
  5   25   125
  6   36   216
```

**"5": Вывести квадраты и кубы 10 чисел следующей последовательности: 1, 2, 4, 7, 11, 16, …**

**Пример:**

```
   1      1      1
   2      4      8
   4     16     64
        ...
  46   2116 97336
```

# Программирование на языке Паскаль

## Циклы с условием

# Цикл с неизвестным числом шагов

**Пример:** Отпилить полено от бревна. Сколько раз надо сделать движения пилой?

**Задача:** Ввести целое число (<2000000) и определить число цифр в нем.

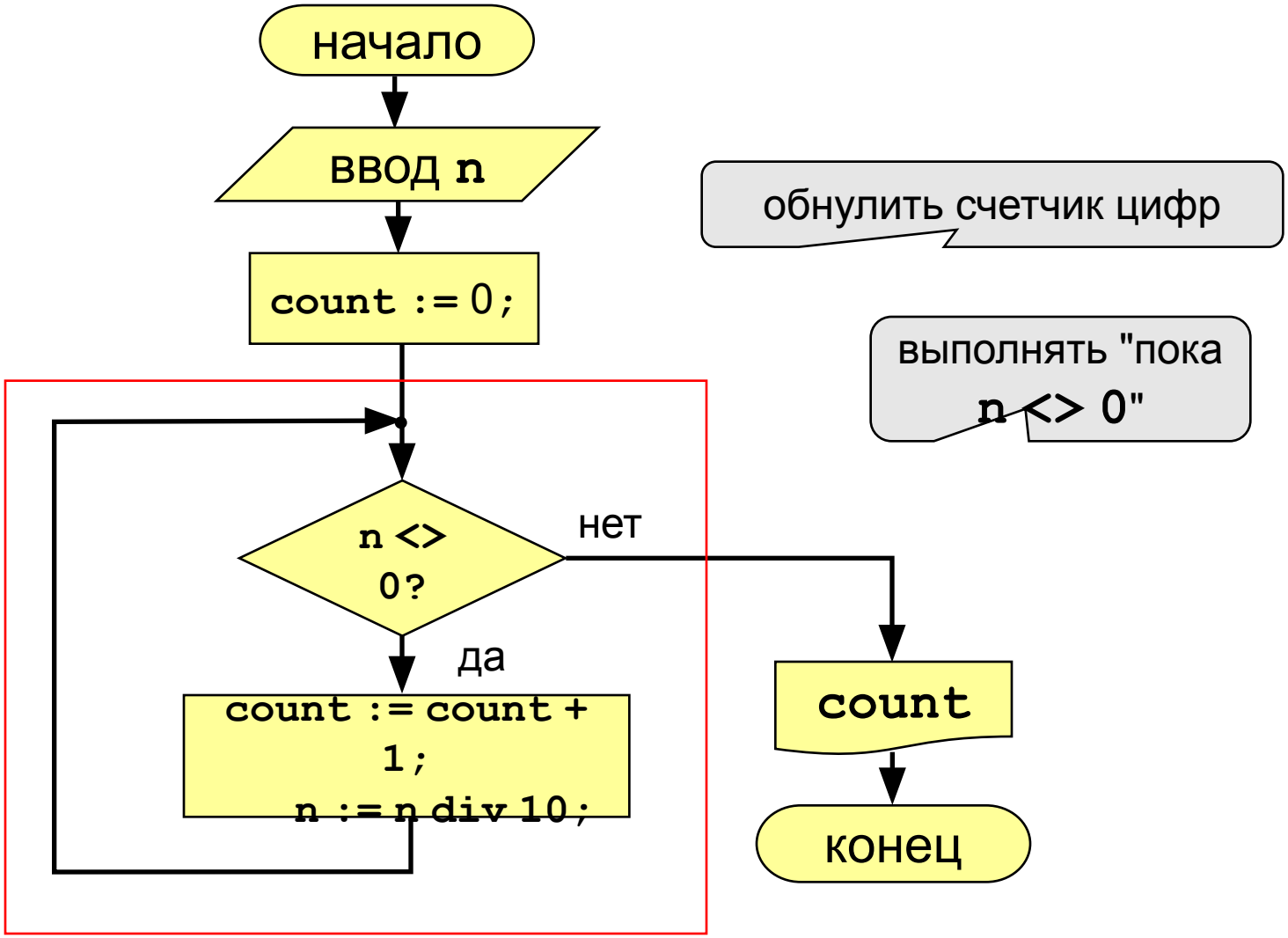
**Идея решения:** Отсекаем последовательно последнюю цифру, увеличиваем счетчик.

| 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 qq;
var n, count, n1: integer;
begin
  writeln('Введите целое число');
  read(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');
end.
```

выполнять "пока n <> 0"

Что плохо?

# Цикл с условием

```
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;
```

# Цикл с условием

**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;
```

# 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:=1 to 10 do begin
  {тело цикла}
end;
```

```
i := 1;
while i <= 10 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.**

# Задания

**"4": Ввести целое число и найти сумму его цифр.**

**Пример:**

```
Введите целое число:
1234
Сумма цифр числа 1234 равна 10.
```

**"5": Ввести целое число и определить, верно ли, что в его записи есть две одинаковые цифры.**

**Пример:**

```
Введите целое число:
1234
Нет.
```

```
Введите целое число:
1224
Да.
```

# Последовательности

**Примеры:**

- 1, 2, 3, 4, 5, …
- 1, 2, 4, 7, 11, 16, …
- 1, 2, 4, 8, 16, 32, …
- $\frac{1}{2}, \frac{1}{2}, \frac{3}{8}, \frac{1}{4}, \frac{5}{32}, \ldots$

$a_n = n$

$a_1 = 1,\ a_{n+1} = a_n + 1$

$a_1 = 1,\ a_{n+1} = a_n + n$

$a_n = 2^{n-1}$

$a_1 = 1,\ a_{n+1} = 2a_n$

$\frac{1}{2}, \frac{2}{4}, \frac{3}{8}, \frac{4}{16}, \frac{5}{32}, \ldots$

$a_n = \frac{b_n}{c_n}$

$b_1 = 1,\ b_{n+1} = b_n + 1$

$c_1 = 2,\ c_{n+1} = 2c_n$

# Последовательности

**Задача:** найти сумму всех элементов последовательности,

$$1,\ -\frac{1}{2},\ \frac{2}{4},\ -\frac{3}{8},\ \frac{4}{16},\ -\frac{5}{32},\ldots$$

которые по модулю больше 0,001:

$$S = 1 - \frac{1}{2} + \frac{2}{4} - \frac{3}{8} + \frac{4}{16} - \frac{5}{32} + \ldots$$

**Элемент последовательности (начиная с №2):**

$$a = z\,\frac{b}{c}$$

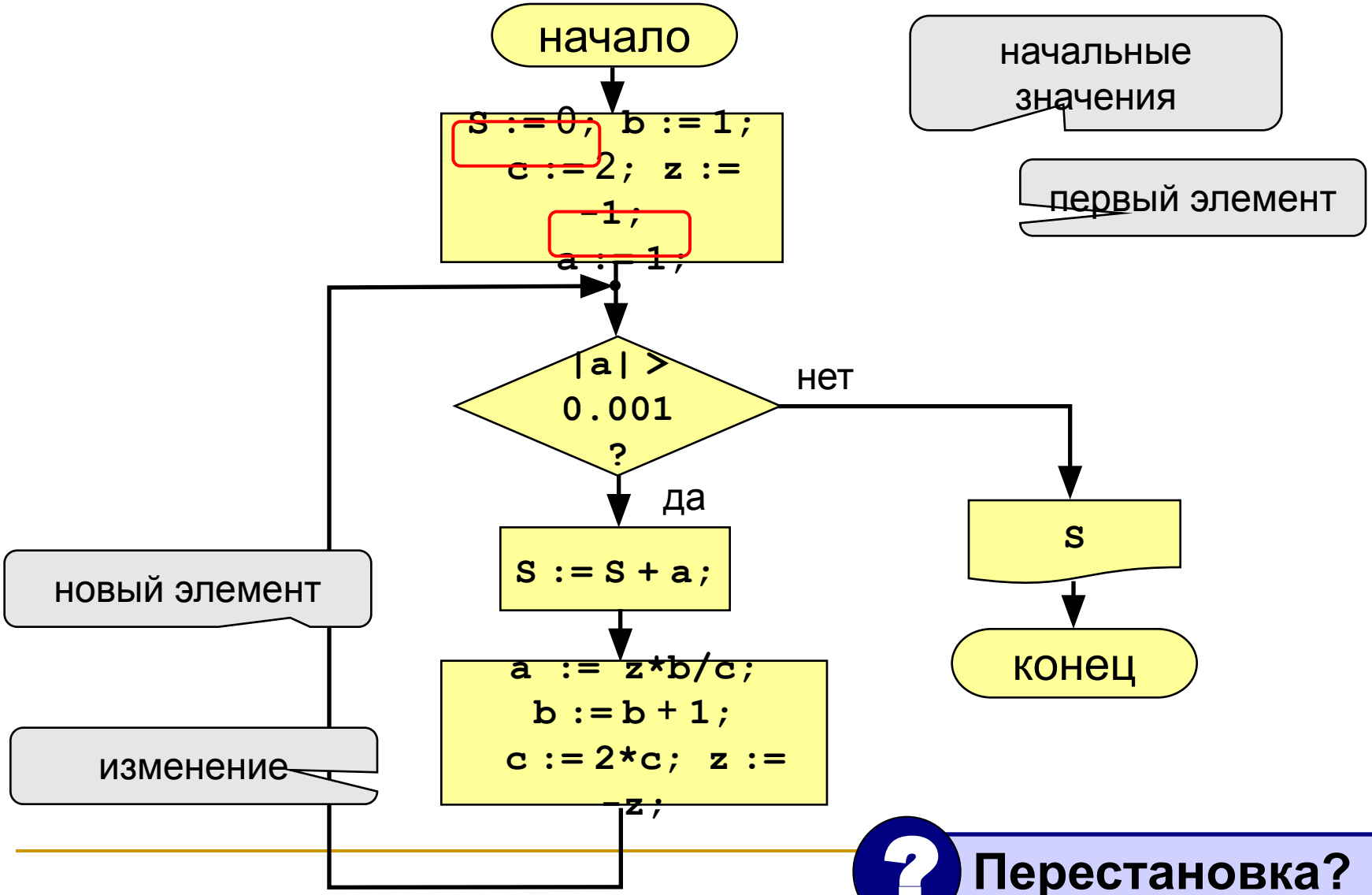
| n | 1 | 2 | 3 | 4 | 5 | ... |
|---|---|---|---|---|---|---|
| b | 1 | 2 | 3 | 4 | 5 | ... |
| c | 2 | 4 | 8 | 16 | 32 | ... |
| z | -1 | 1 | -1 | 1 | -1 | ... |

```
b := b+1;
```

```
c := 2*c;
```

```
z := -z;
```

# Алгоритм




**Перестановка?**

# Программа

```
program qq;
var b, c, z: integer;
    S, a: real;
begin
   S := 0; z := -1;
   b := 1; c := 2; a := 1;
   while abs(a) > 0.001 do begin
      S := S + a;
      a := z * b / c;
      z := - z;
      b := b + 1;
      c := c * 2;
   end;
   writeln('S =', S:10:3);
end.
```

начальные значения

увеличение суммы

расчет элемента последовательности

переход к следующему слагаемому

# Задания

**"4": Найти сумму элементов последовательности с точностью 0,001:**

$$S = 1 + \frac{2}{3 \cdot 3} - \frac{4}{5 \cdot 9} + \frac{6}{7 \cdot 27} - \frac{8}{9 \cdot 81} + \ldots$$

**Ответ:**

```
S = 1.157
```

**"5": Найти сумму элементов последовательности с точностью 0,001:**

$$S = 1 + \frac{2}{2 \cdot 3} - \frac{4}{3 \cdot 9} + \frac{6}{5 \cdot 27} - \frac{8}{8 \cdot 81} + \frac{10}{13 \cdot 243} - \ldots$$

**Ответ:**

```
S = 1.220
```

# Цикл с постусловием

**Задача:** Ввести целое **положительное** число (<2000000) и определить число цифр в нем.

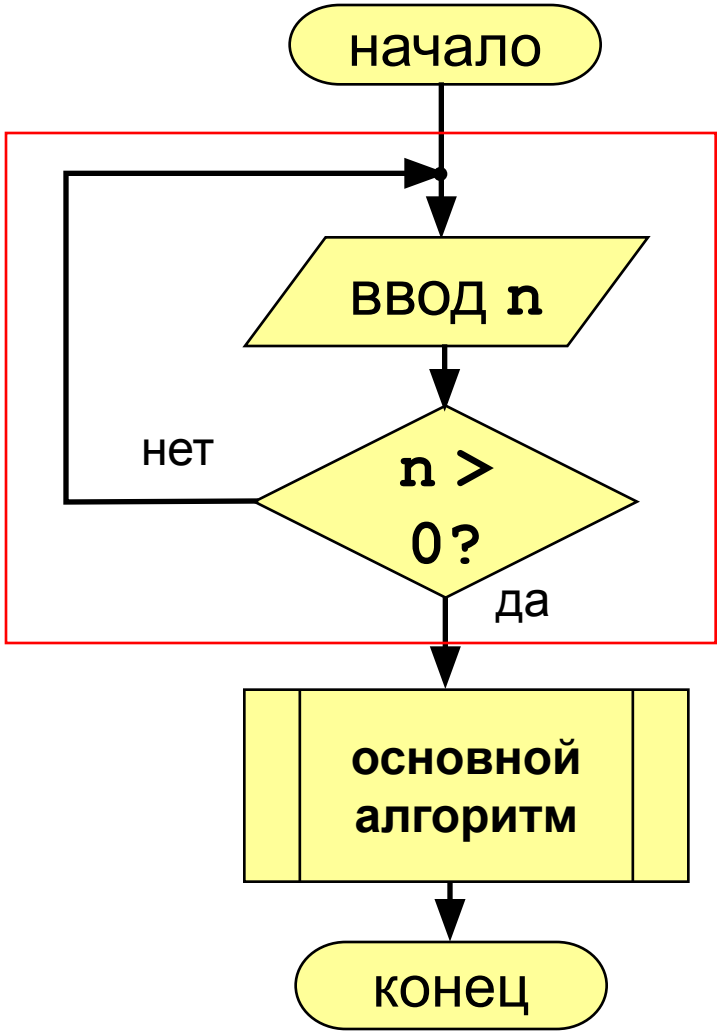
**Проблема:** Как не дать ввести отрицательное число или ноль?

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

**Особенность:** Один раз тело цикла надо сделать в любом случае => проверку условия цикла надо делать в конце цикла (цикл с **постусловием**).

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

# Цикл с постусловием: алгоритм




тело цикла

условие **ВЫХОДА**

блок "типовой процесс"

# Программа

```
program qq;
var n: integer;
begin
  repeat
    writeln('Введите положительное число');
    read(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