



# ПРОГРАММИРОВАНИЕ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ

## НАЧАЛА ПРОГРАММИРОВАНИЯ

9 класс

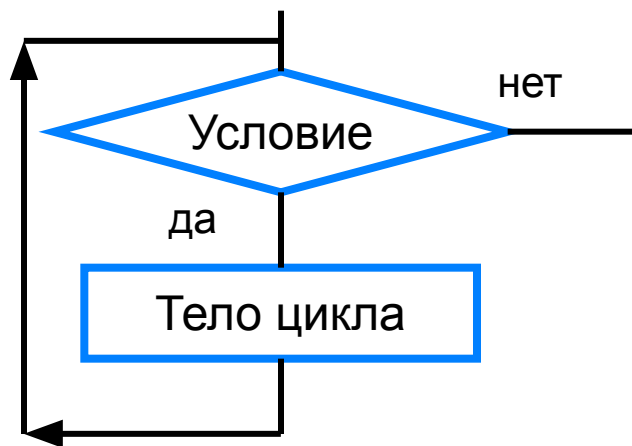
# Ключевые слова

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



# Программирование циклов с заданным условием продолжения работы

Цикл с предусловием, «пока».



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.

- Цикл выполняется до тех пор, пока истинно условие

```
While (x<0) do  
  x:=x+1;
```

- Если несколько операторов в теле цикла, то использовать **BEGIN .... END**

```
While (i<100) do  
  begin  
    writeln('i=',i);  
    i:=i+1;  
  end;
```

# Особенности цикла `while`:

- 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;
```

- 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;
```

# Алгоритм Евклида

Наибольший общий делитель двух натуральных чисел (НОД) – это самое большое натуральное число, на которое они делятся нацело.

$$\text{НОД}(12, 18) = 6$$

## Постановка задачи

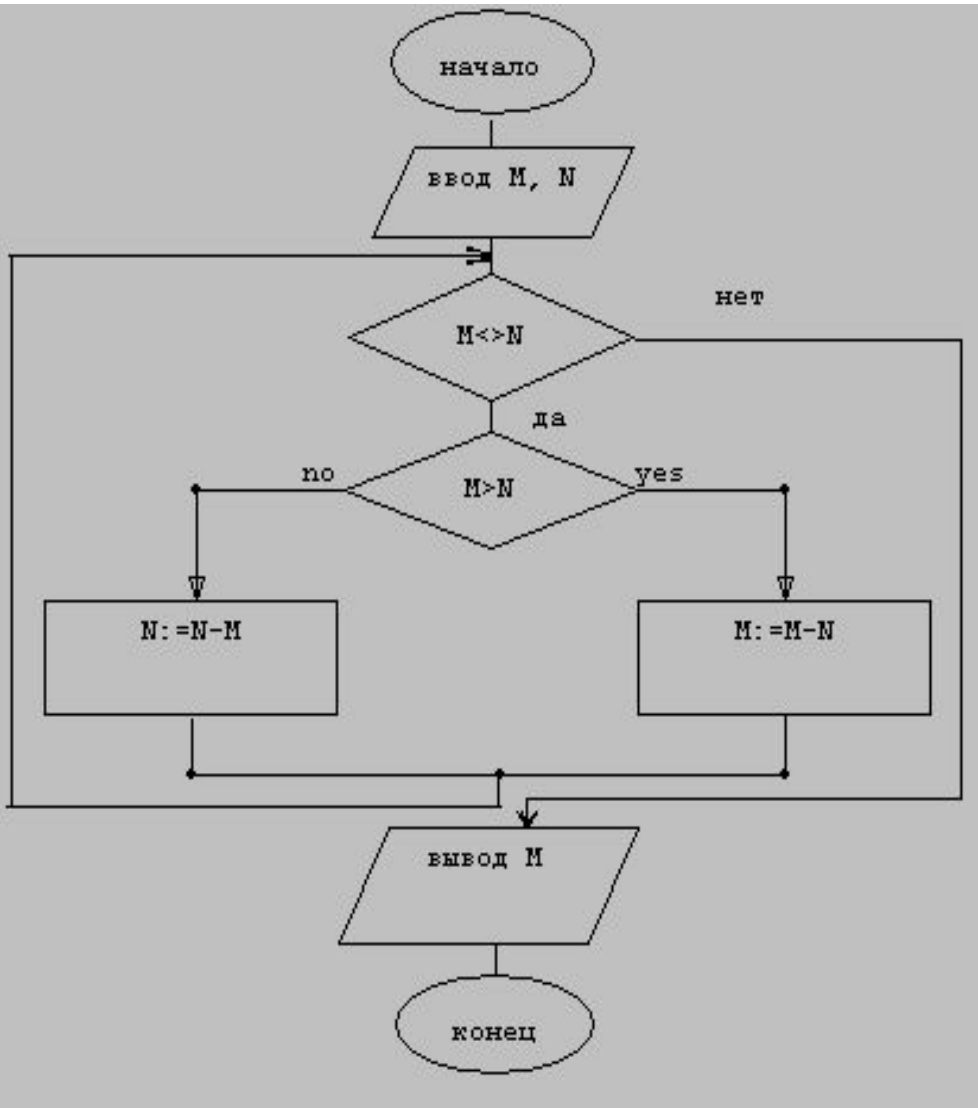
Дано:  $M, N$

Найти:  $\text{НОД}(N, M)$

## Решение задачи

Из большего числа вычесть меньшее до тех пор пока числа не будут равны.

# Блок-схема



```
Program Evklid;  
Var M, N: integer;
```

```
Begin
```

```
  Writeln('Введите M и N');  
  Readln(M, N);
```

```
  While M > N do
```

```
    begin
```

```
      if M > N then M:=M-N  
      else N:=N-M
```

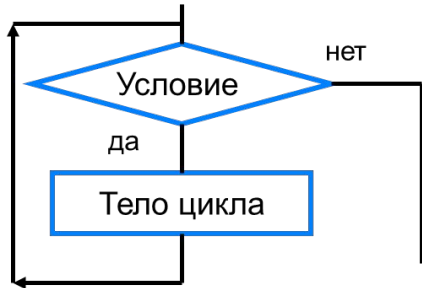
```
    end;
```

```
  Write('НОД=', M);
```

```
  Readln
```

```
End.
```

# Задание 1. Сколько раз выполняется цикл?



а)  $a := 4; b := 6;$   
*while*  $a < b$  *do*  $a := a + 1;$

б)  $a := 4; b := 6;$   
*while*  $a < b$  *do*  $a := a + b;$

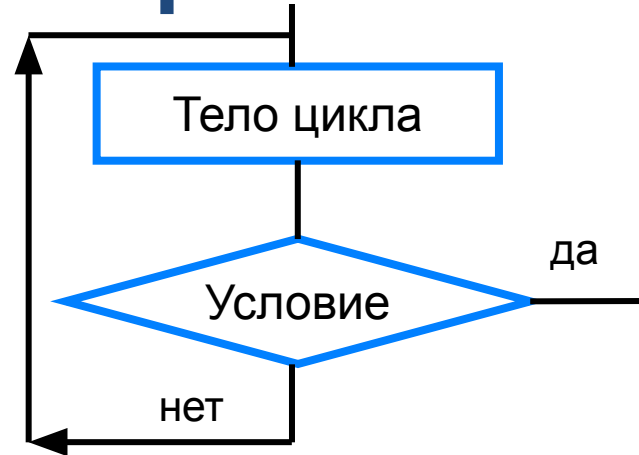
в)  $a := 4; b := 6;$   
*while*  $a > b$  *do*  $a := a + 1;$

г)  $a := 4; b := 6;$   
*while*  $a < b$  *do*  $b := a - b;$

д)  $a := 4; b := 6;$   
*while*  $a < b$  *do*  $a := a - 1;$



# 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.

# Цикл с постусловием, цикл «ДО»

**Операторы REPEAT ... UNTIL**  
Формат оператора:

**REPEAT {повторять}**

**<Тело цикла>**

**UNTIL условие ; {до тех пор, пока не}**

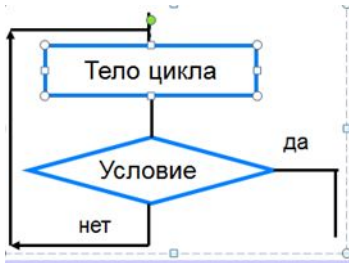
**Цикл выполняется пока условие не  
станет истинным!**

- **Особенности цикла repeat:**
  - • можно использовать сложные условия
  - • цикл выполняется хотя бы один раз
  - • условие пересчитывается каждый раз при выходе из цикла
  - • цикл заканчивает выполнение когда условие становится верным
  - • если условие на выходе цикла всегда ложно, программа зацикливается
  -

**Задача . Ввести натуральное число и  
определить, верно ли, что сумма его цифр  
равна 10.**

```
program qq;  
var n, a, s: integer;  
Begin  
  read(n); s:=0;  
  repeat  
    a:= n mod 10;  
    s:=s+a;  
    n:=n div 10;  
  until n = 0;  
  if s=10 then writeln('верно');  
end.
```

## Задание 2. Сколько раз выполняется цикл?



а)  $a := 4; b := 6;$   
*repeat*  $a := a + 1; \text{until } a > b;$

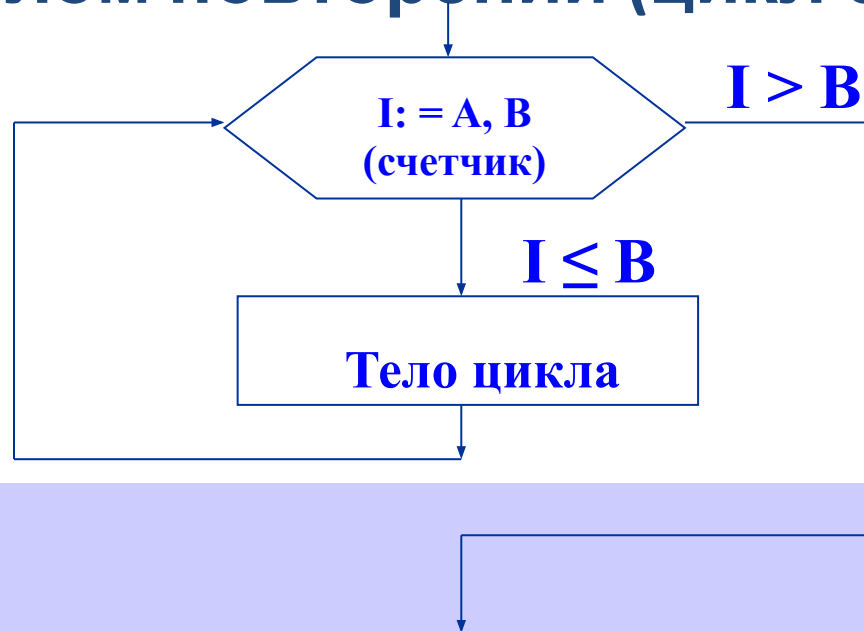
б)  $a := 4; b := 6;$   
*repeat*  $a := a + b; \text{until } a > b;$

в)  $a := 4; b := 6;$   
*repeat*  $a := a + b; \text{until } a < b;$

г)  $a := 4; b := 6;$   
*repeat*  $b := a - b; \text{until } a < b;$

д)  $a := 4; b := 6;$   
*repeat*  $a := a + 2; \text{until } a < b;$

# Программирование циклов с заданным числом повторений (цикл со счетчиком)



Общий вид оператора:

```
for <параметр>:=<начальное_значение>  
to <конечное_значение> do <оператор>
```

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

# Цикл с увеличением

параметра

**FOR** I:= A **TO** B **DO**

<Тело цикла>;

I – параметр, увеличивается на 1

A – начальное значение

B – конечное значение

Тело цикла – один оператор!!!

Или – скобки **BEGIN ... END;**



# Цикл с уменьшением параметра

- **FOR I:=A DOWNTO B DO**  
**<ТЕЛО ЦИКЛА> ;**



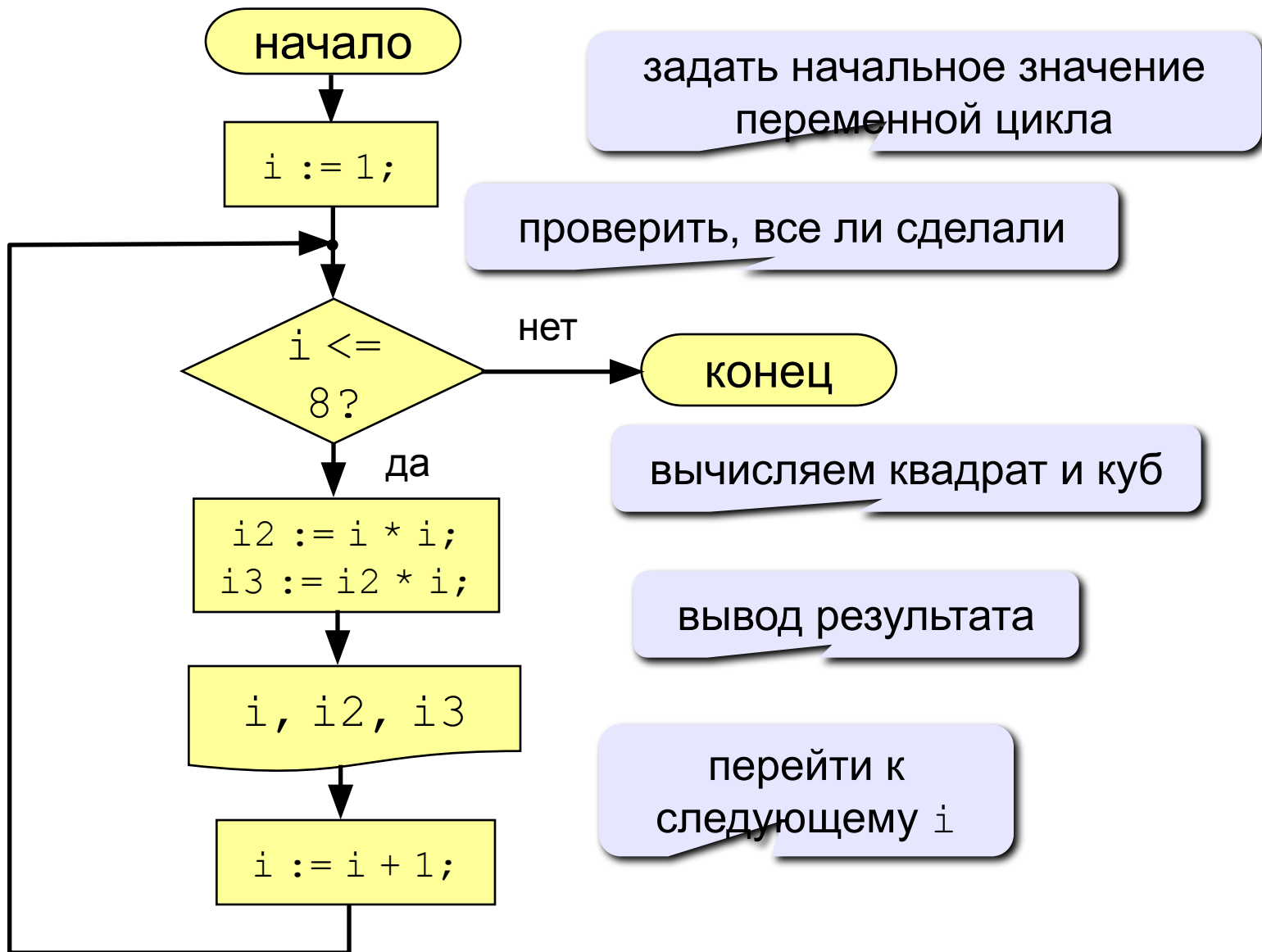
- выполняется аналогичным образом, но значение параметра уменьшается на 1.



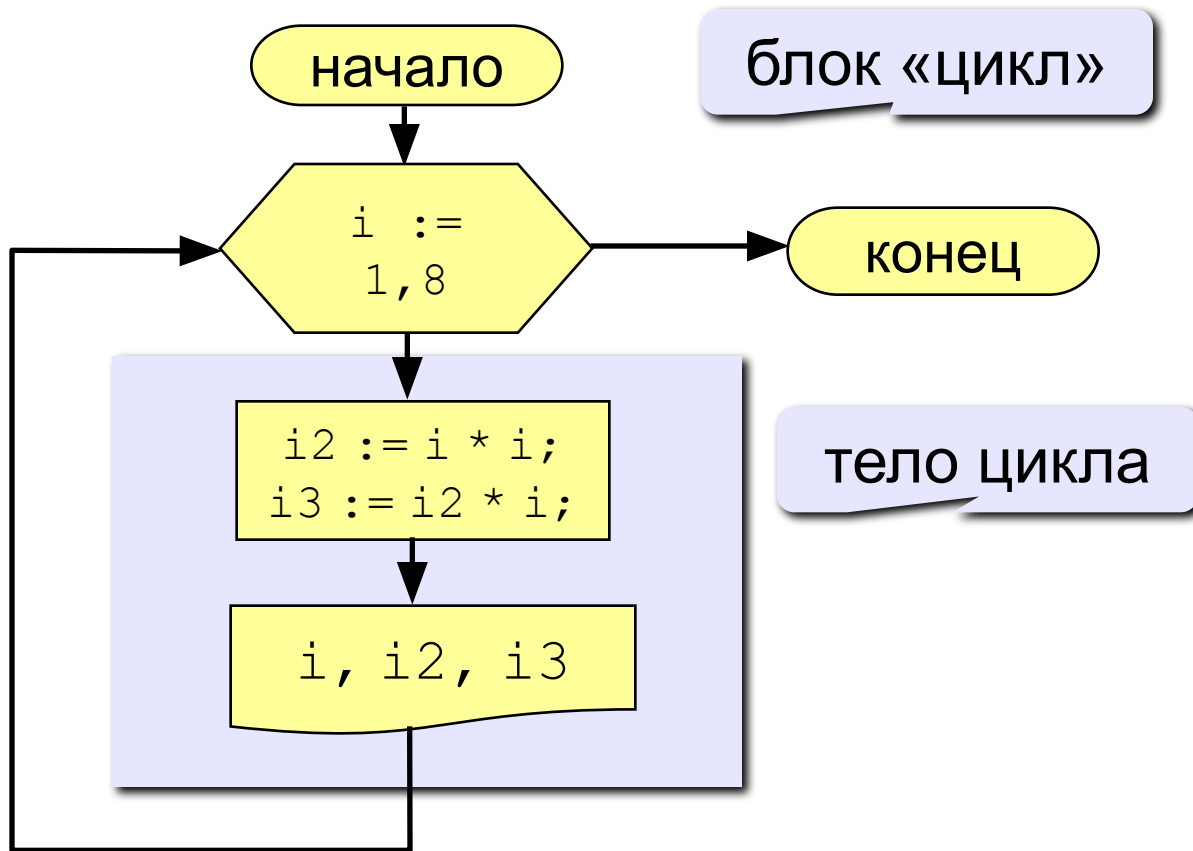
**Задача.** Вывести на экран квадраты и кубы  
целых чисел от 1 до 8 (от a до b).

```
i := 1;      { очередное число }  
i2 := i*i;   { его квадрат }  
i3 := i2*i;  { куб }  
writeln(i, i2:4, i3:4);  
i := 2;
```

# Алгоритм



# Алгоритм (с блоком «цикл»)



# Программа

```
program qq;  
var i, i2, i3: integer;
```

начальное значение

переменная  
цикла

конечное значение

```
    for i:=1 to 8 do begin  
        i2 := i*i;  
        i3 := i2*i;  
        writeln(i:4, i2:4, i3:4);  
    end;  
end.
```

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

# Цикл с переменной

---

## 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)

Количество итераций цикла определяется разностью между вторым и первым значением плюс единица.

---

```
    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

## Цикл с предусловием

*while* <условие> *do begin*

    {тело цикла}

*end;*

## Особенности цикла **while**:

- 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*

$a := a + 1;$

- условие пересчитывается **каждый раз** при входе в цикл
- если условие на входе в цикл ложно, цикл не выполняется ни разу
- если условие никогда не станет ложным, программа

**зацикливается**



## Цикл с постусловием

*repeat*

*{тело цикла}*

*until* <условие>;

### Особенности цикла **repeat**:

- можно использовать сложные условия
- цикл выполняется **хотя бы один раз**
- условие пересчитывается **каждый раз** при выходе из цикла
- цикл заканчивает выполнение когда условие становится верным
- если условие на выходе цикла **всегда** ложно, программа **зацикливается**

# №7. Найти сумму всех положительных целых чисел, не превышающих данного натурального числа N.

## Цикл с постусловием

```
Program Summa2;  
Var S, N, a: integer;  
Begin  
  Writeln('Введите N');  
  Readln(N);  
  a:=1; S:=0;  
  Repeat  
    S:=S+a;  
    a:=a+1;  
  Until a>N  
  Write('сумма =', S);  
  Readln  
End.
```

## Цикл с параметром

```
Program Summa3;  
Var S, N, a: integer;  
Begin  
  Writeln('Введите N');  
  Readln(N);  
  S:=0;  
  For a:= 1 to N do  
    S:=S+a;  
  Write('сумма =', S);  
  Readln  
End.
```

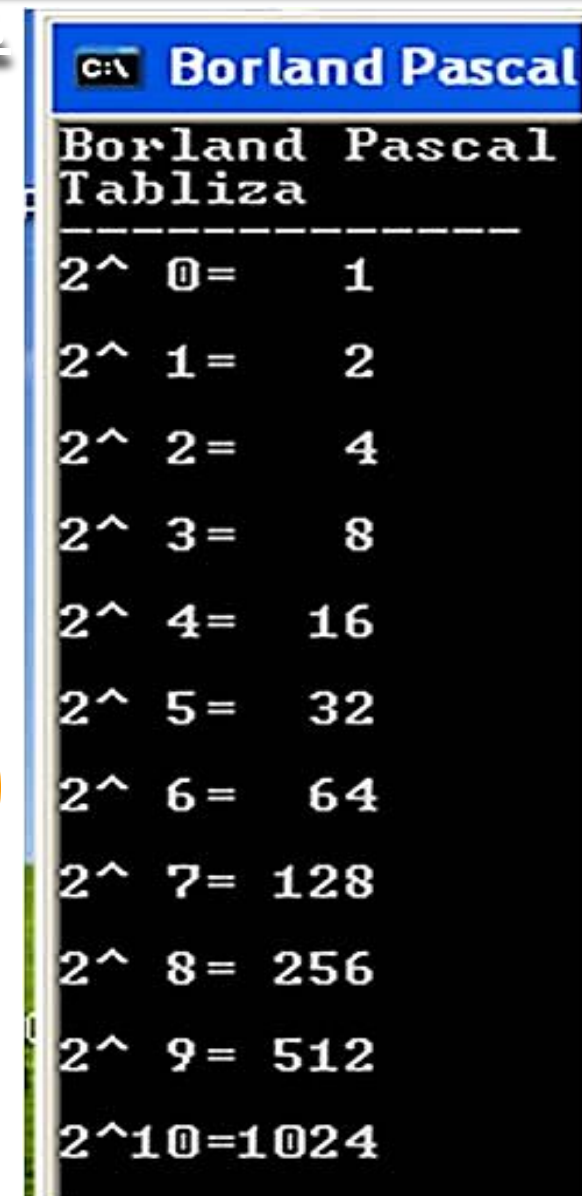
### Задание 3. Определить результат выполнения фрагмента программы:

**a)**    **s: =0; i: =1;**  
**repeat**  
**s:= s +10 Div i;**  
**i:=i+1;**  
**until l>5;**

**б)**    **a: =1; b: =1;**  
**While a <= 4 Do**  
**begin**  
      **a: =a + 1;**  
      **b: = b + 5;**  
**end;**

Задача 2. Написать программу, выводящую на экран степени числа 2 (от 0 до 10) в виде

```
Program stepen2;
Var i, x: integer;
Begin
  Writeln ('Tabliza');
  Writeln ('-----');
  x :=1;
  For i := 0 to 10 do
    Begin
      Writeln ('2^', i:2, '=', x:5);
      x := x*2;
      Writeln;
    End;
  Readln;
End.
```



```
c:\ Borland Pascal
Borland Pascal
Tabliza
-----
2^ 0=  1
2^ 1=  2
2^ 2=  4
2^ 3=  8
2^ 4= 16
2^ 5= 32
2^ 6= 64
2^ 7= 128
2^ 8= 256
2^ 9= 512
2^10=1024
```

# Вопрос

1	2	3	4
2	4	6	8
3	6	9	12
4	8	12	16

1. Можно ли в теле цикла `for` использовать цикл `for`?

```
...  
Так For i := 1 to 4 do  
    Begin
```

2. Что будет результатом выполнения следующего кода?  

```
        For j := 1 to 4 do  
            Write( i*j, ' ');  
        Writeln;  
    End;
```

3. Как ...

а какой – внутренним?

ВНЕШНИМ.

кране в  
ующего  
?

ешним,

# Алгоритм Евклида

*Алгоритм Евклида* – это алгоритм нахождения наибольшего общего делителя (НОД) двух целых неотрицательных чисел.

Пусть  $x$  и  $y$  одновременно не равные нулю целые неотрицательные числа и пусть  $x \geq y$ .

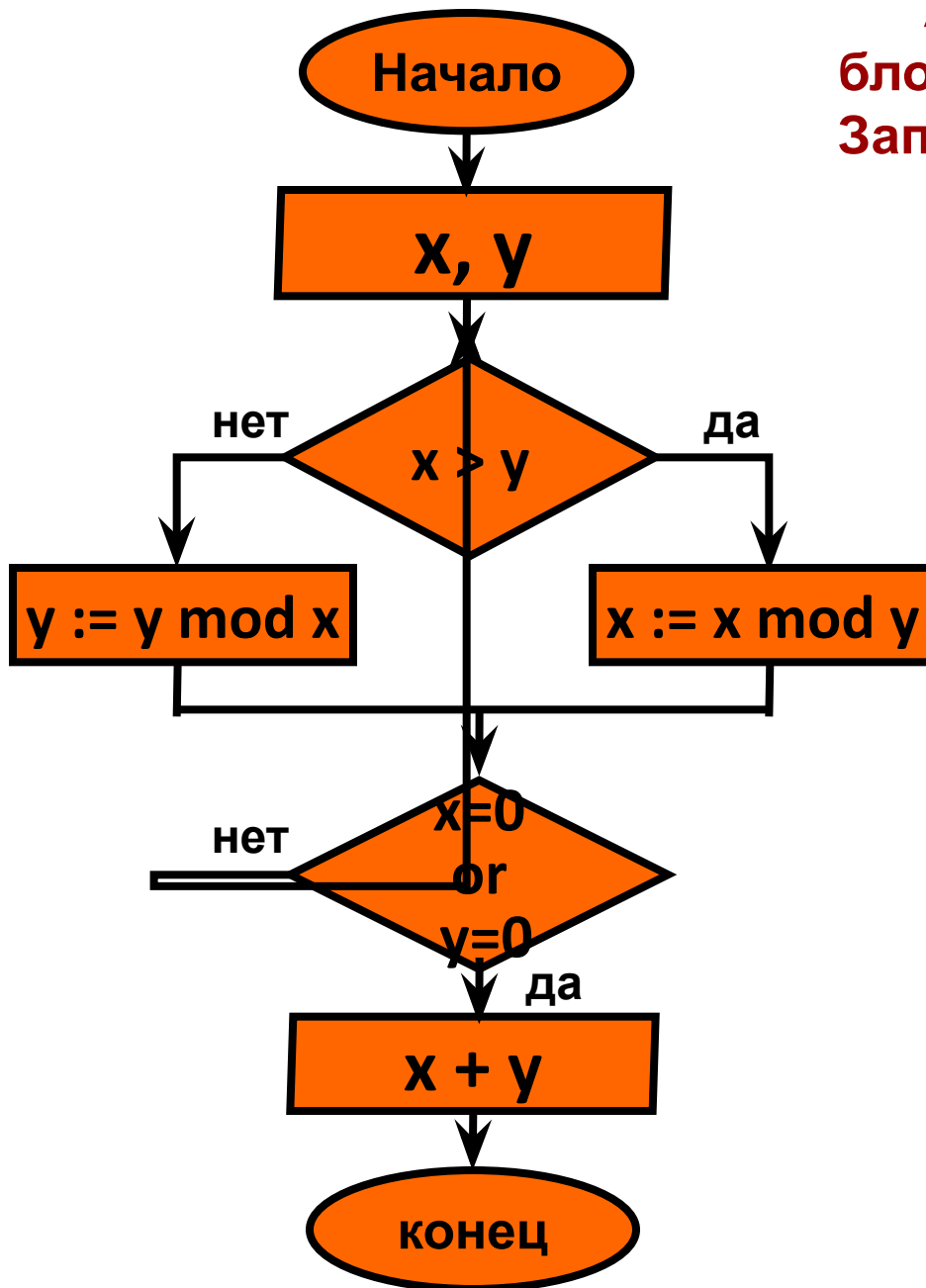
Если  $y = 0$ , то  $\text{НОД}(x, y) = x$ , а если  $y \neq 0$ , то для чисел  $x$ ,  $y$ , и  $r$ , где  $r$  – остаток от деления  $x$  на  $y$ , выполняется равенство

$$\text{НОД}(x, y) = \text{НОД}(y, r).$$

Например, пусть  $x = 48$ , а  $y = 18$ .

$$\text{НОД}(48, 18) = \text{НОД}(18, 12) = \text{НОД}(12, 6) =$$

Алгоритм Евклида изображен  
блок-схемой «цикл с постусловием»  
Запишите его на языке Turbo Pascal.



```
Program NOD;
```

```
Var x, y : integer;
```

```
Begin
```

```
Write('vvod x, y');
```

```
Readln (x, y);
```

```
Repeat
```

```
if x > y then x := x mod y
```

```
else y := y mod x
```

```
Until (x = 0) or (y = 0);
```

```
Writeln ('NOD=', x + y);
```

```
Readln;
```

```
End.
```

# Решение задач



Получить таблицу температур по Цельсию от 0 до 100 градусов и их эквивалентов по шкале Фаренгейта, используя для перевода формулу:

$$t_f = \frac{9}{5}t_c + 32$$

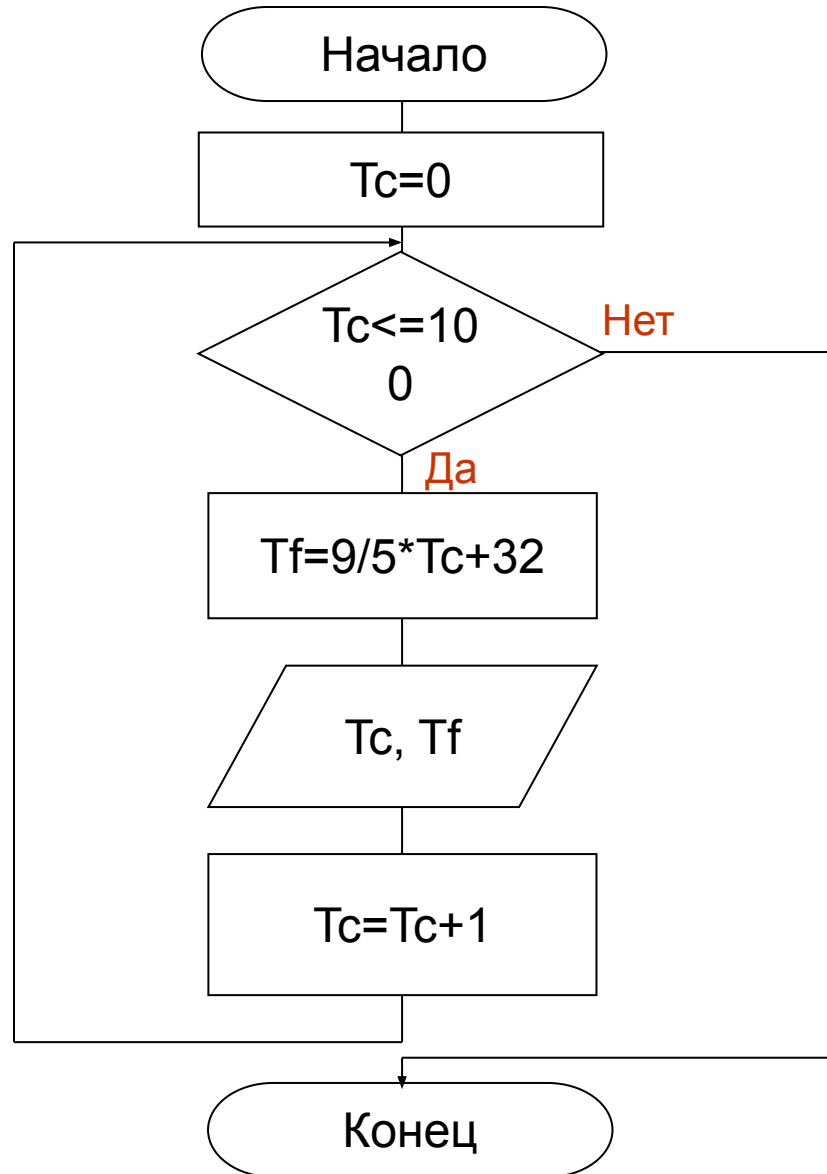
Решить задачу тремя способами, используя операторы:

- While While (цикл с предусловием) While (цикл с предусловием)
- Repeat Repeat (цикл с постусловием)
- For For (цикл со счётчиком)



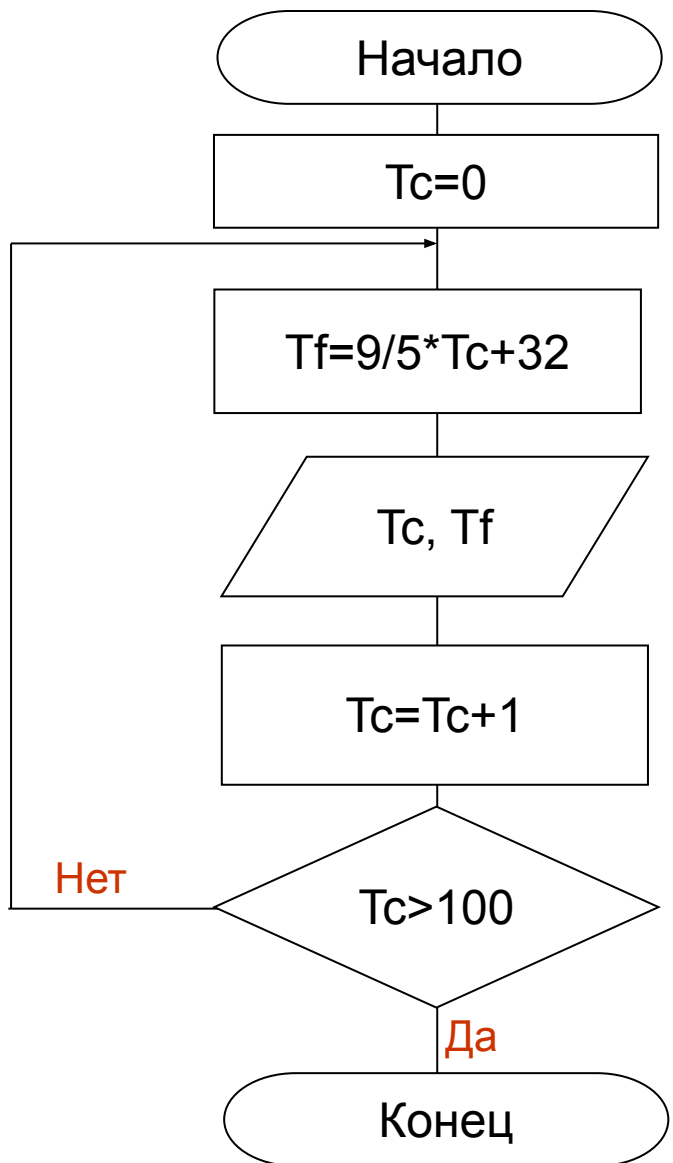


# While (цикл с предусловием)



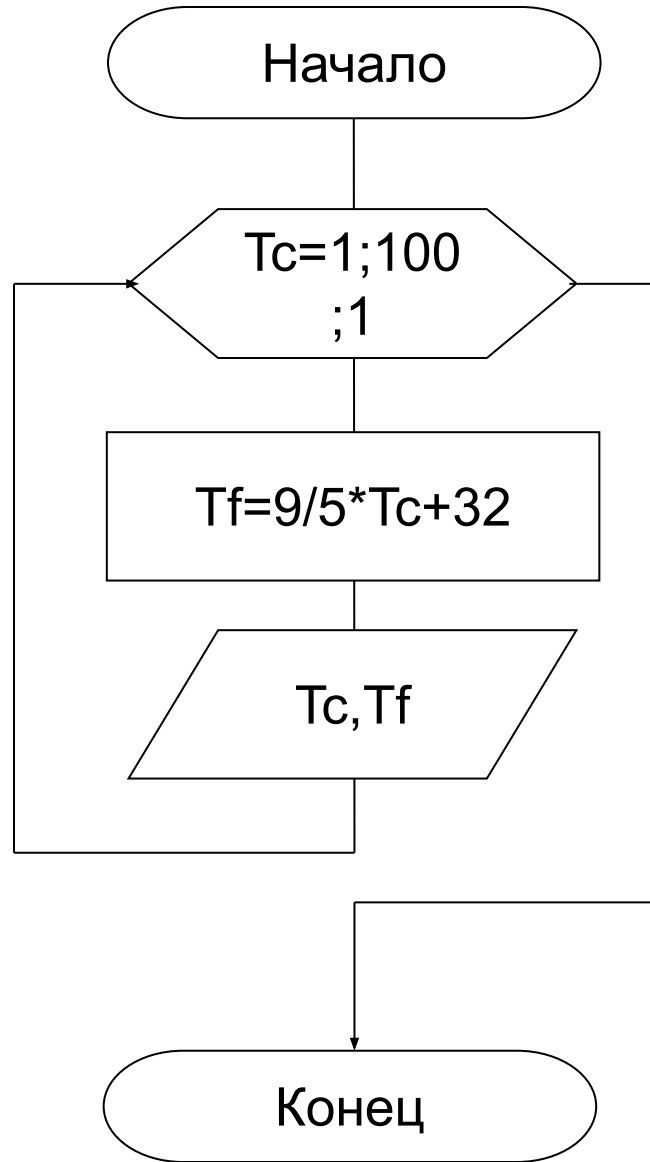


# Репеат (цикл с постусловием)





# For (цикл со счётчиком)



# 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.

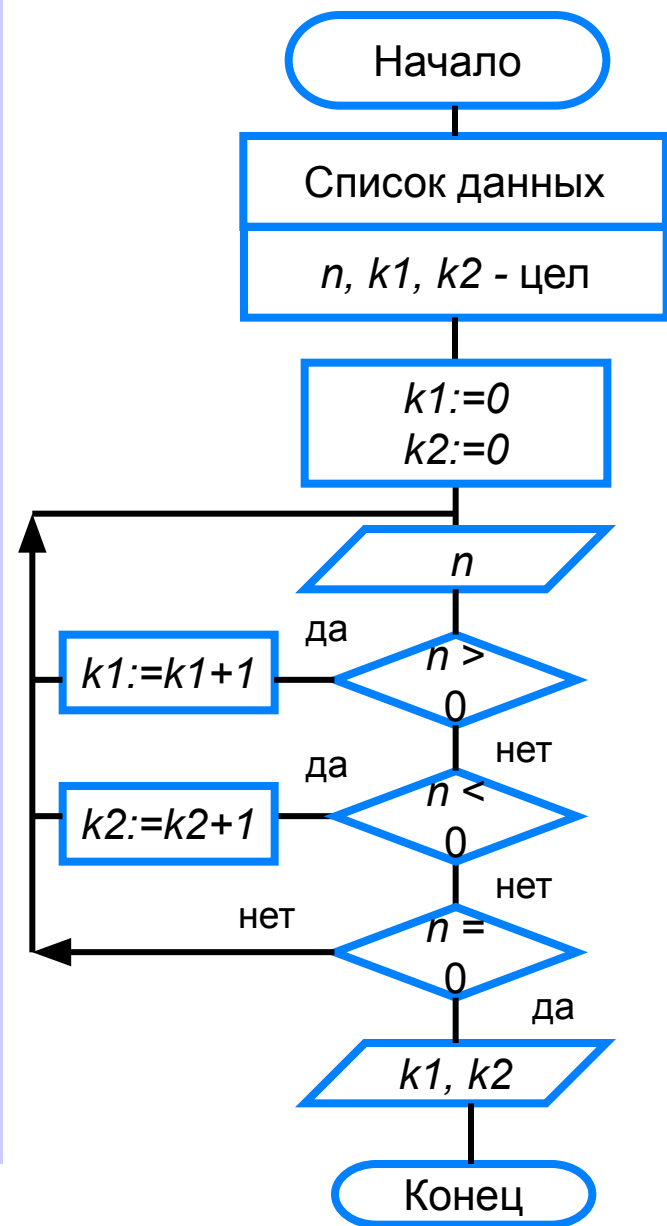


Воспользуемся оператором **repeat**.

```

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 введен ноль.

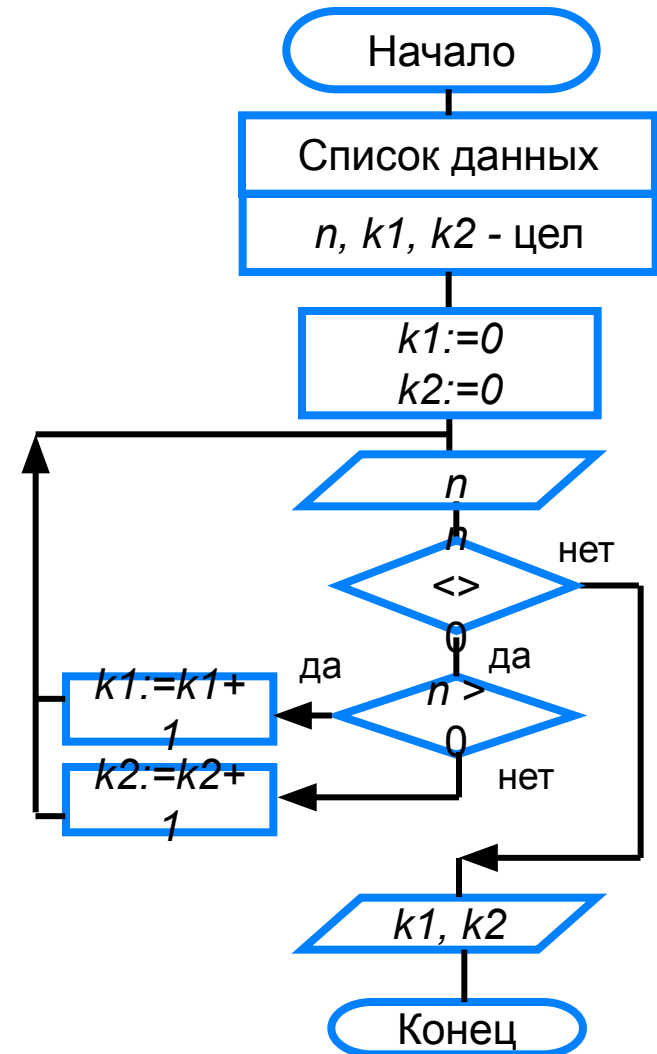


Работа продолжается, пока  $n \neq 0$ .



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

Напишите программу, которая выводит на экран таблицу значений факториала для чисел от 1 до n. Используйте оператор цикла. Выводите результат в виде таблицы.

Какой из трёх рассмотренных операторов цикла является, по вашему мнению, основным, и каким, что является, по вашему мнению, основным, данными

именно в том, что операторы цикла не работают с числами. Обобщите результат оператора repeat

Сколько раз будет повторен цикл; какими будут значения переменных a, b, c. Найдите ошибки. Допишите необходимые операторы. этой последовательности операторов? Сколько раз будет повторен цикл; какими будут

Пример входных данных	Пример выходных данных
Введите n > 5	$5 \times 7 = 35$ $5 \times 8 = 40$ $5 \times 9 = 45$
Введите n > 6	$5 \times 10 = 50$



# Опорный конспект

В языке Паскаль имеются три вида операторов цикла:

*for* (цикл с параметром).

Число повторений  
цикла известно

*repeat* (цикл-ДО)

Число повторений  
цикла неизвестно

*while* (цикл-ПОКА)