Операторы цикла

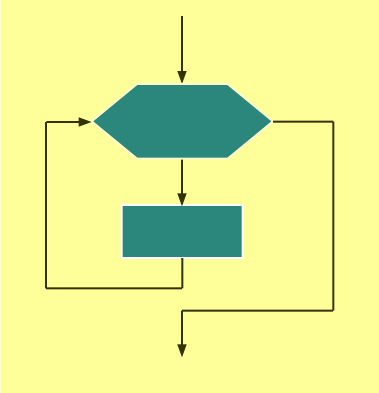
Циклы позволяют многократно выполнять одну или группу команд, причем в тексте программы нет необходимости записывать эти команды несколько раз.

В языке программирования PASCAL существует три вида циклов:

1. **Арифметический (перечисляемый) цикл FOR**
2. **Логический цикл с предусловием WHILE**
3. **Логический цикл с послеусловием REPEAT**

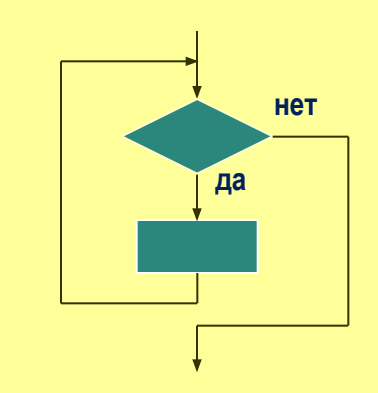
Циклические операторы на Pascal

1. Цикл с параметром



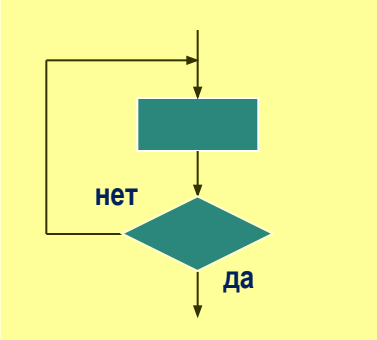
for счетчик:=значение to конечное_значение do тело_цикла;

2. Цикл с предусловием (пока)



WHILE *x<10* DO *s:=s+x*;

3. Цикл с постусловием (до)



REPEAT n:=n+1 UNTIL f<9;

Оператор после do повторяется до тех пор, пока логическое условие истинно

Оператор цикла FOR

Часто цикл **for** называют циклом со счетчиком. Этот цикл используется, когда число повторений не связано с тем, что происходит в теле цикла. Т.е. количество повторений может быть вычислено заранее (хотя оно не вычисляется).

Цикл **for** существует в двух формах:

for счетчик:=значение to конечное_значение do тело_цикла;

for счетчик:=значение downto конечное_значение do тело_цикла;

Если между начальным и конечным выражением указано служебное слово **to**, то на каждом шаге цикла значение параметра будет *увеличиваться на единицу*. Если же указано **downto**, то значение параметра будет *уменьшаться на единицу*.

Прямой пересчет идет от известного меньшего числа до известного большего, на каждом шаге прибавляется единица (например, от 20 до 25: 20, 21, 22, 23, 24, 25).
Обратный пересчет – от большего к меньшему. И на каждом шаге вычитается единица.

1) цикл с прямым отсчетом:

```
for i := n1 to n2 do оператор;
```

2) цикл с обратным отсчетом:

```
for i := n2 downto n1 do оператор;
```

Арифметический цикл FOR

```
For i := n1 to n2 do оператор;
```

i – параметр цикла (счетчик), переменная целого типа;
n1 и n2 – начальное и конечное значения счетчика.

Особенностью арифметического цикла является то, что число повторений операторов цикла должно быть известно заранее. Решение о выполнении или невыполнении в очередной раз тела цикла принимается до начала его прохождения, поэтому может случиться так, что тело цикла не будет выполнено ни разу.

Порядок выполнения цикла FOR

1) вычисляются значения выражений n1, n2;
2) параметру цикла присваивается значение n1;
3) если полученное значение счетчика больше n2, то выполнение цикла заканчивается;
4) выполняется тело цикла;
5) значение параметра цикла увеличивается на 1, осуществляется переход к пункту 3.

Количество проходов цикла с заголовком

```
for i := n1 to n2 do
```

можно вычислить по формуле n2 - n1 + 1

Сколько раз будет выполнено тело цикла с данным заголовком?

```
1) for i := -10 to -4 do
2) for i := 6 to 2 do
3) for i := 3 to 3 do
```

Цикл под номером 1 будет выполняться для счетчика, последовательно принимающего значения –10, –9, –8, –7, –6, –5, –4,то есть 7 раз.

Цикл под номером 2 не будет выполняться ни разу, так как начальное значение счетчика больше конечного.

Цикл под номером 3 будет выполняться 1 раз для счетчика, равного 3.

Значение счетчика цикла может использоваться в выражениях, входящих в операторы тела цикла, но изменение значения счетчика цикла этими операторами недопустимо.

Попробуйте ответить, какие фрагменты программы записаны без ошибок.

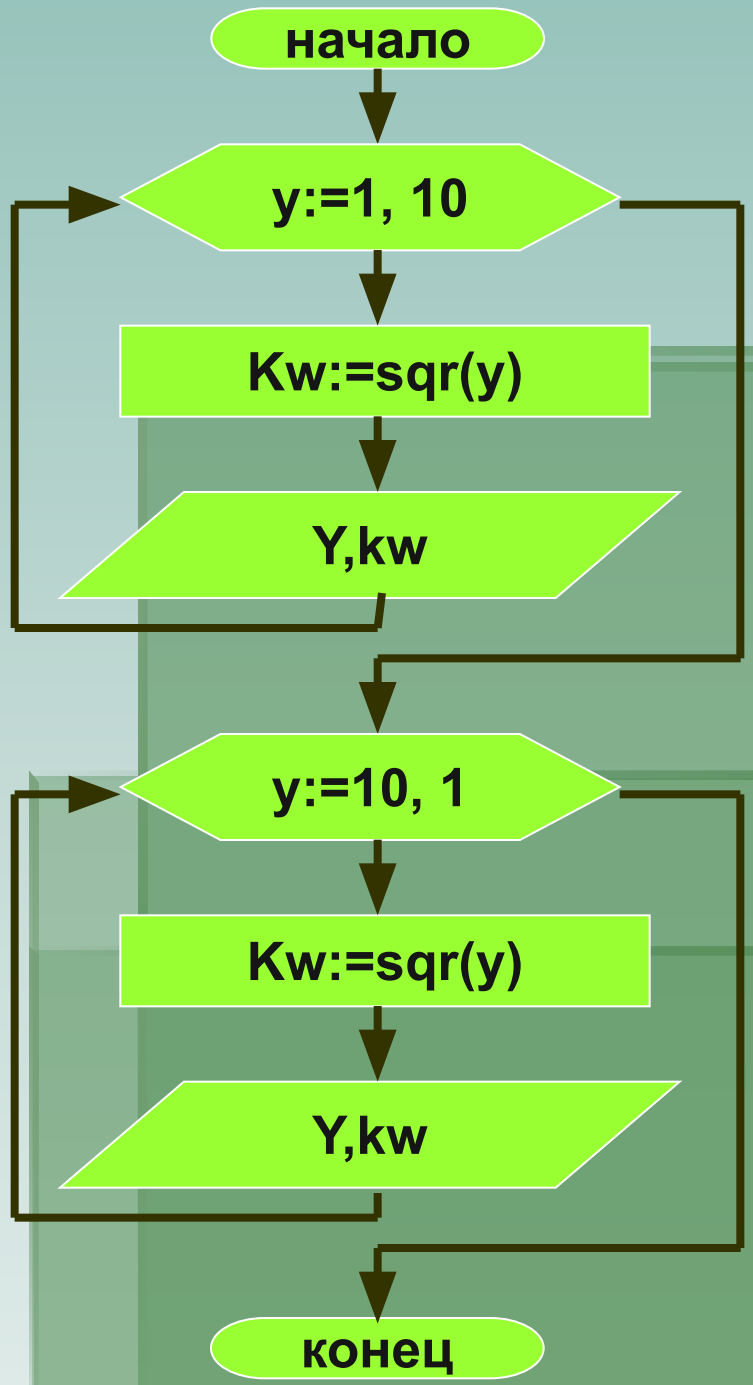
```
1) FOR I := 2 TO 20 Do
   WriteLn (I);
2) FOR I := 2 TO 20 Do
   I := I + 1;
3) FOR I:= 2 TO 20 Do
   ReadLn (I);
4) FOR I := 2 TO 20 Do
   A := I * I;
```

Правильно записаны фрагменты под номерами 1 и 4.

Во фрагментах 2 и 3 в операторах тела цикла производится изменение значение счетчика цикла.

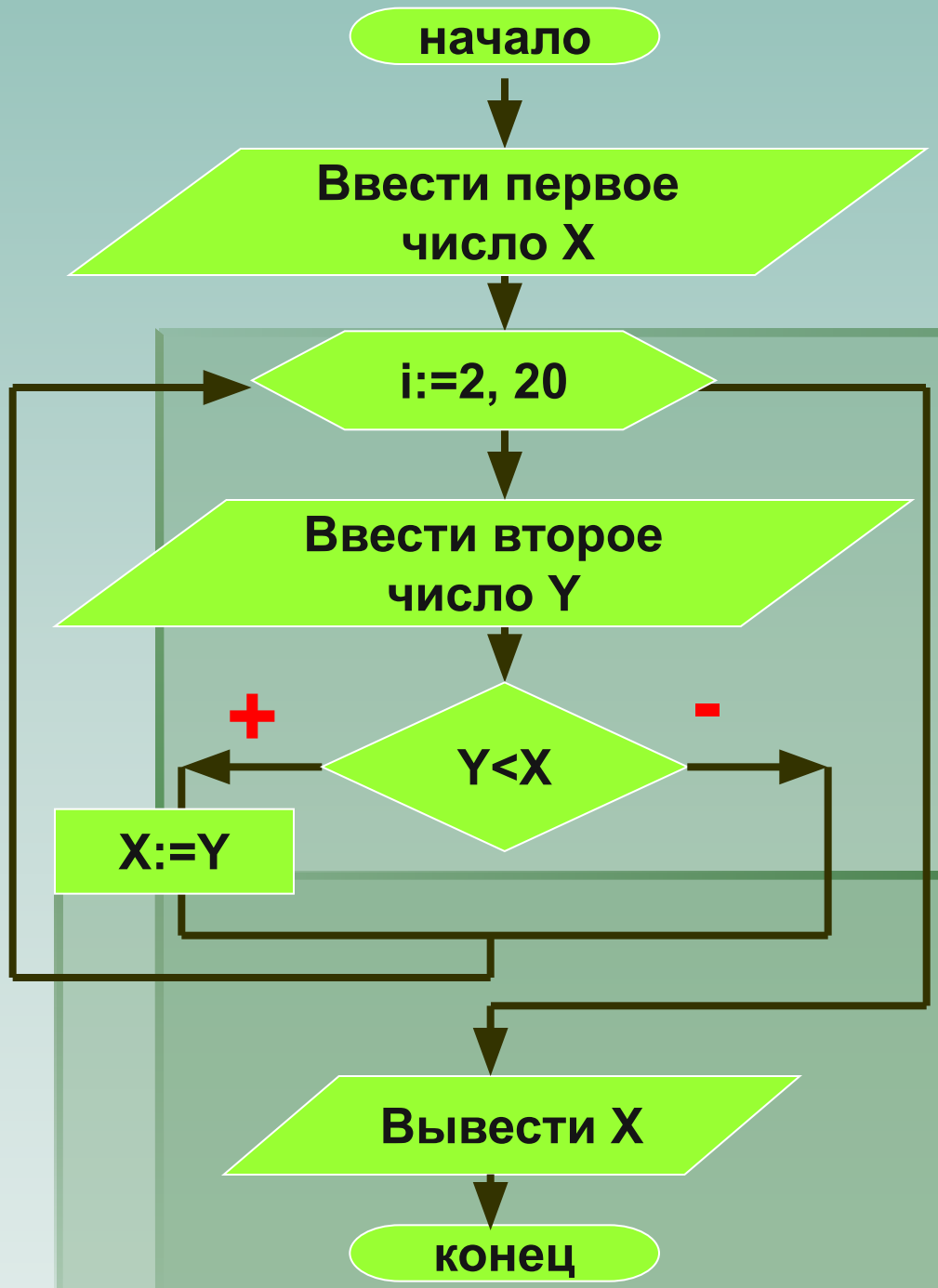
Примеры использования оператора цикла FOR

Вывести на экран значение y^2 ($y=1,2,...,10$) в возрастающем и убывающем порядке




```
Program KWADRAT;
Uses crt;
Var kw, y: integer;
Begin
Clrscr;
For y:=1 to 10 do
 Begin
       kw:= sqr(y);
       Writeln (y:3,kw:5);
   End;
Writeln;
For y:=10 downto 1 do
   Begin
        kw:=sqr(y);
        Writeln (y:3,kw:5);
    End;
Readln;
End.
```

Выбрать наименьшие значение из 20 чисел, вводимых с клавиатуры




```
program min;
uses crt;
var i:integer;
x,y : real;
begin
clrscr;
   writeln ('Enter first');
    readln (x);
      for i:=2 to 20 do
        begin
            write ('Enter next');
             readln (y);
         if y<x then x:= y
      end;
        writeln ('min=',x:6:2);
readln;
end.
```

Вычисление суммы и количества чисел

Вычислить сумму n чисел, вводимых с клавиатуры.

```
program Summa;
uses crt;
var I, n :integer;
x, sum : real;
begin
clrscr;
writeln (‘Enter  n’);
readln (n);
 sum:=0;
 for i:=1 to n do
        begin
             write (‘Enter  x’);
             readln (x);
        sum:=sum+ x;
        end;
writeln (‘sum=’, sum:6:2);
readln;
end.
```

Среди всех двузначных чисел найти те, сумма цифр которых равна n (0<n≤18) и количество этих чисел.

```
Program Chisla;
uses crt;
var n, i,  k, p1, p2:integer;
begin
clrscr;
k:=0;
writeln (‘Vvesti  n’);
readln (n);
for i:=10 to 99 do
        begin
            p1:=i div 10;
            p2:= i mod 10;
            if (p1+ p2) =n then
                begin
                    writeln (i);
                    k:=k+1;
                end;
    end;
 writeln (‘Koli4estvo 4isel ’, k);
readln;
end.
```

Логические циклы

При составлении программ часто возникают ситуации, когда

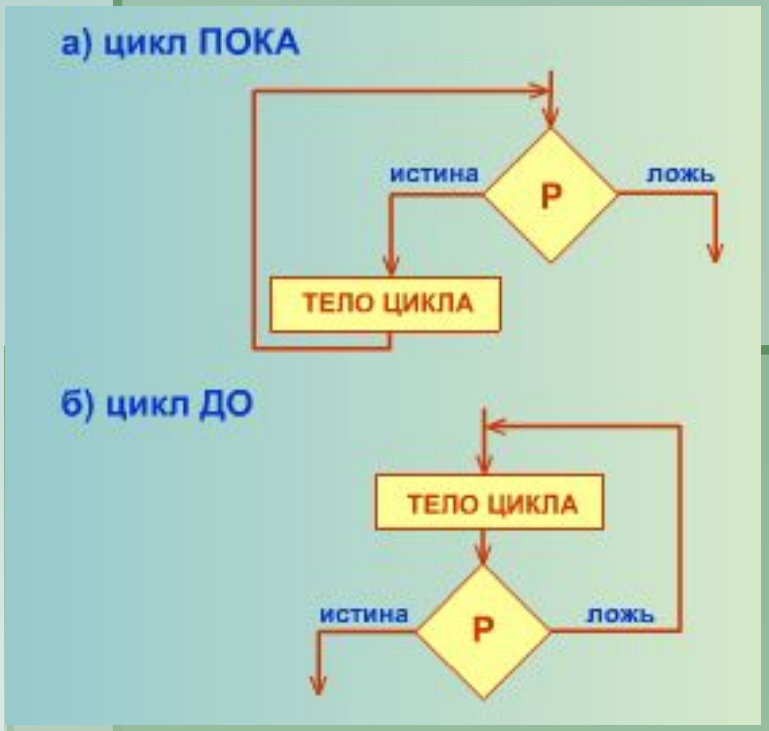
1. Заранее не известно количество повторений цикла;
2. Переменная – счетчик цикла должна изменяться с шагом ≠ 1.

В таких случаях используют

Логические циклы

Логические циклы

- Это циклическое повторение блока команд, пока выполняется (или не выполняется) некоторое условие



1) цикл с предусловием (ПОКА):

```
while условие do
  <оператор>;
```

2) цикл с постусловием (ДО):

```
repeat
  <операторы>
until условие;
```

Сравнение циклов While и Repeat

Цикл ПОКА	Цикл ДО
while условие do <оператор>;	repeat <операторы> until условие

1. В цикле **While** проверка условия выполнения цикла находится в начале цикла, а в **Repeat** – в конце. Цикл **Repeat** всегда выполняется хотя бы один раз, а цикл **While** может не выполнятся ни разу.
2. В цикле **While** выход из цикла осуществляется, если условие ложно, а в **Repeat** – если условие истинно.
3. Между словами **Repeat** и **Until** можно размещать несколько операторов без **Begin** и **End**, а цикл **While…do** может содержать только один оператор тела цикла;

Оператор с предусловием While

Оператор while (пока) называют оператором цикла с предусловием за то, что проверка условия выполнения тела цикла производится в самом начале оператора, до операторов тела цикла. Если условие изначально не выполнится, то операторы тела цикла не выполнятся ни разу.

Формат оператора:

While <условие выполнения цикла> do begin <тело цикла>; end;

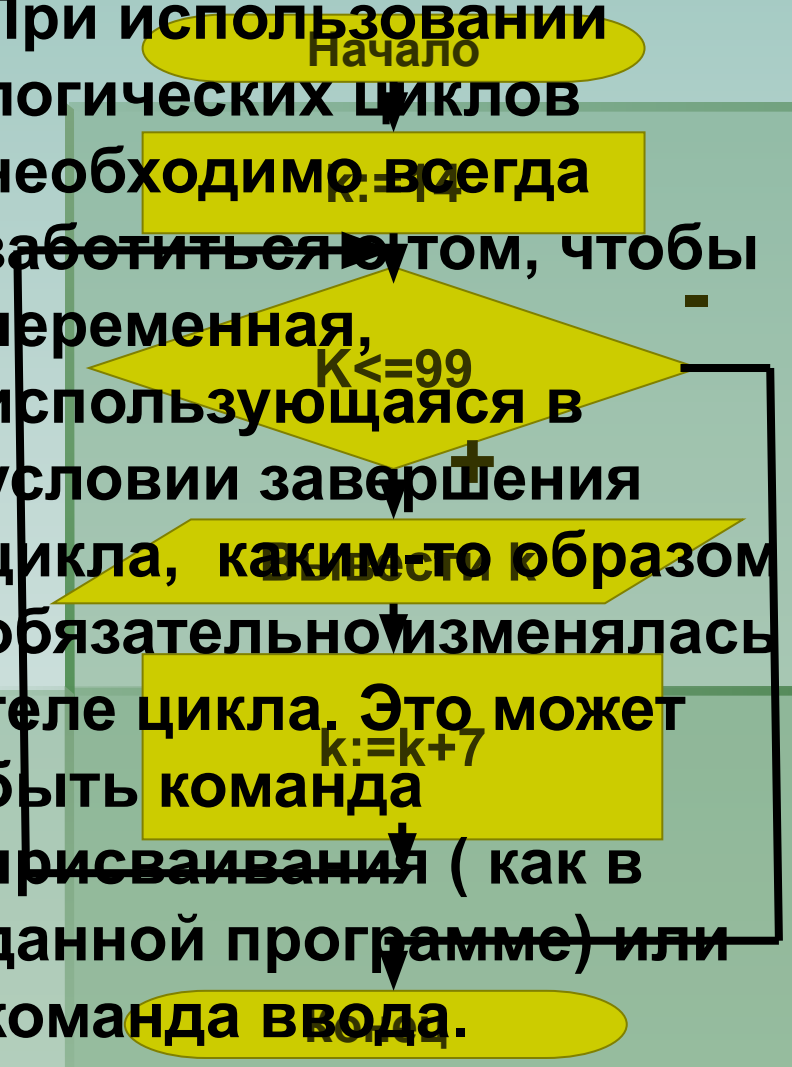
Здесь: <условие выполнения цикла> - булевское выражение; <тело цикла> - операторы, которые будут повторяться.

На русском языке это звучит примерно так:

пока выполняется условие делай
начало
<тело цикла>
конец

Найти все двузначные числа, кратные 7.

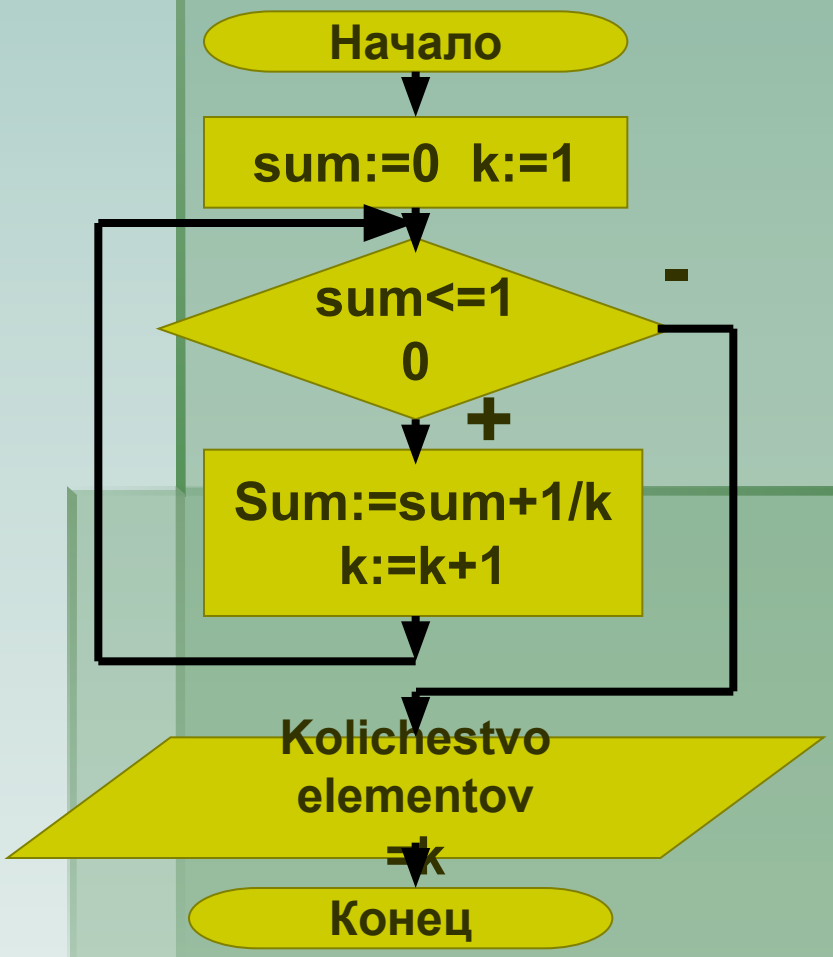
При использовании логических циклов необходимо всегда заботиться о том, чтобы переменная, использующаяся в условии завершения цикла, каким-то образом обязательно изменялась в теле цикла. Это может быть команда присваивания (как в данной программе) или команда ввода.




```
Program Kratn_7;
Uses Crt;
Var k: integer;
Begin
Clrscr;
k:=14;
While k<=99 do
    Begin
        Writeln(k);
        k:=k+7;
    end;
readln;
end.
```

При каком количестве слагаемых последовательности их сумма превысит 10?

$$1+\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{k}$$




```
Program Summa_2;
Uses Crt;
Var sum, x: real;
    K: integer;
Begin
Clrscr;
sum:=0;
k:=1;
While sum<=10 do
     Begin
          Sum:=sum+1/k;
          k:=k+1;
     end;
Writeln('Kolichestvo elementov=',k);
readln;
end.
```

Использование цикла Repeat

Оператор цикла repeat аналогичен оператору **while**, **но** отличается от него, **во-первых**, тем, что условие проверяется после очередного выполнения тела цикла (за это и называется циклом с постусловием) и таким образом гарантируется хотя бы однократное выполнение цикла, а **во-вторых**, тем, что выполнение условия (равенство булевского выражения константе true) является критерием не повторения, а прекращения цикла.

Формат оператора:

Repeat <оператор 1> ; <оператор 2> ; . . . <оператор n>;
Until <условие окончания цикла>;

Алгоритм выполнения:

Выполняется тело цикла (операторы, заключенные между словами repeat / until).

Проверяется условие выхода из цикла.

Если условие выполняется, то происходит выход из цикла к первому после repeat оператору.

Если условие не выполняется, то алгоритм повторяется с пункта 1.

Написать программу, которая «задумывает» число в диапазоне от 1 до 9 и предлагает пользователю угадать это число за 5 попыток.

Пояснения к программе:

1. Компьютер может «задумать» число с помощью функции Random.
2. Количество повторений цикла в этой задаче может быть от 1 до 5 – то есть, заранее не известно.
3. Так как пользователь должен сделать хотя бы одну попытку, то логично использовать команду цикла Repeat…until.

```
Program Ugaday_chislo;
const Npop=5;
var comp, igrok, n: integer;
Begin
Randomize;
comp:=random(9)+1;
Writeln ('Game "Угадай число" ');
Writeln (‘Компьютер загадывает число от 1 до 9’);
Writeln (‘Вы должны угадать это число');
Writeln (‘за 5 попыток');
     repeat
          n:=n+1;
          Write (‘Введите число- ');
          readln(igrok);
     until (n=Npop) or (comp=igrok);
     if comp = igrok
         then Writeln (‘Угадал!!!')
         else Writeln (' Не угадал!  Это число - ',comp);
readln;  end.
```

Найти все цифры и их количество заданного произвольного целого числа x>0

Алгоритм:

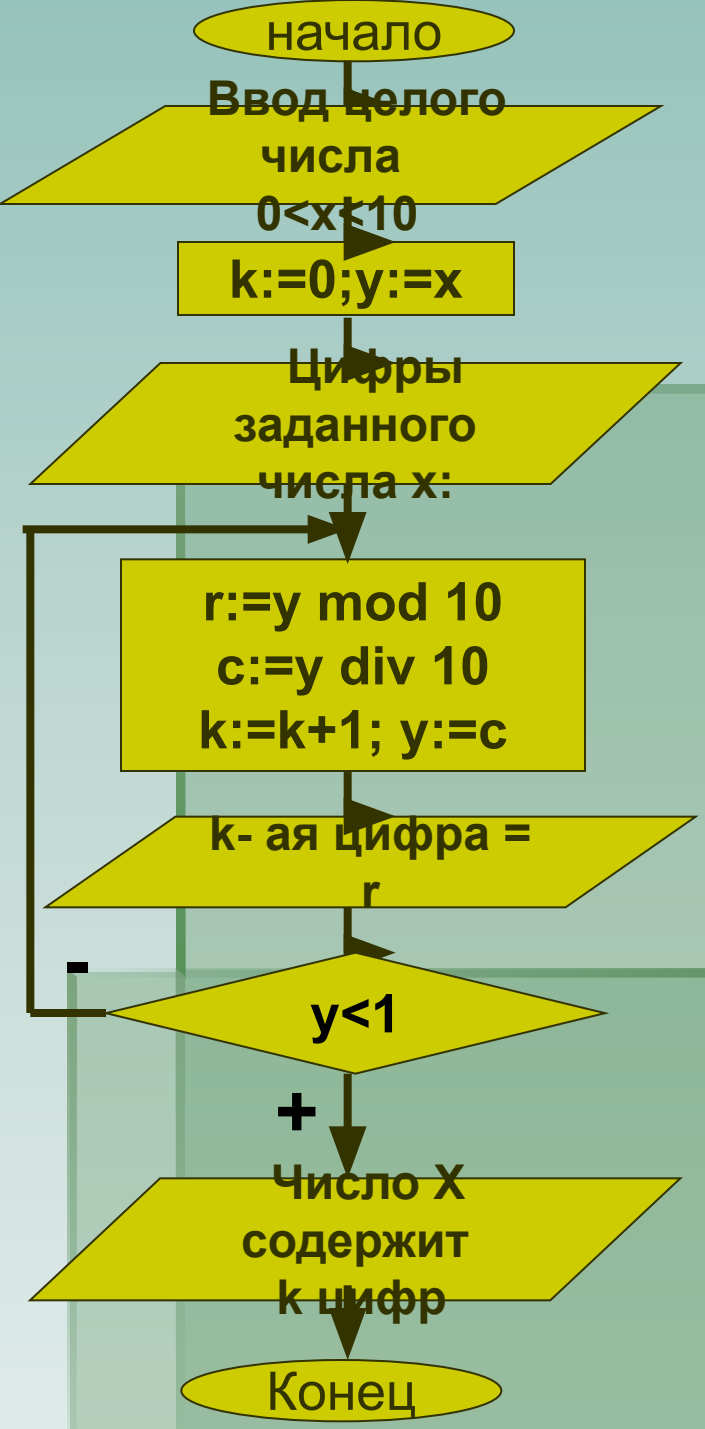
Для того, чтобы не «потерять» заданное число, введем переменную y, значение которой сначала будет =x.

C- целая часть от деления числа y на 10

R- остаток от деления y на 10

K- кол-во цифр в числе x

Последовательно будем находить целую часть и остаток от деления числа **y** на 10.Именно остаток от деления будет очередной цифрой **x.** На каждом шаге будем изменять число **y**: присваивать ему значение целой части от деления **y** на 10.Для нахождения количества цифр организуем счетчик, значение которого будет увеличено на 1.Процесс повторяется пока значение числа **y** не станет меньше 1.




```
Program Zifry;
Uses Crt;
var x, y, c: longint;
   r, k: integer;
Begin
clrscr;
Writeln ('Enter zeloe chislo do 10 snakov');
Readln (x);
writeln;
y:=x;   k:=0;
Writeln ('Zifry zadannogo chisla:');
     repeat
          r:=y mod 10;
          c:=y div 10;
          k:=k+1;
          y:=c;
          Writeln (k,' zifra =  ',r);
     until y<1;
Writeln ('Chislo - ',x,' coderjit ', k, ' zifr');
readln;
end.
```