Алгоритм циклической структуры

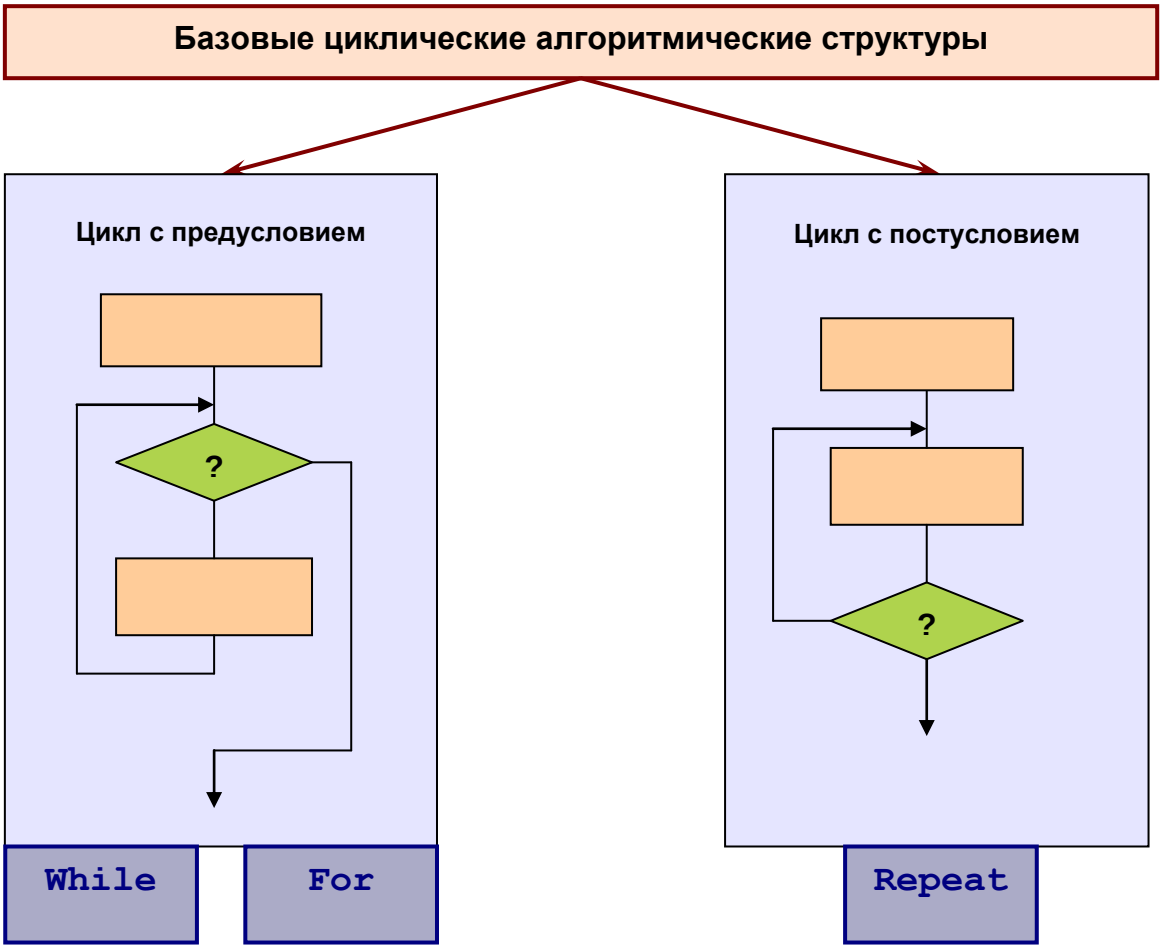
1. **Подготовка цикла** – задание начальных значений переменным цикла перед первым его выполнением.

2. **Тело цикла** – действия, повторяемые в цикле.

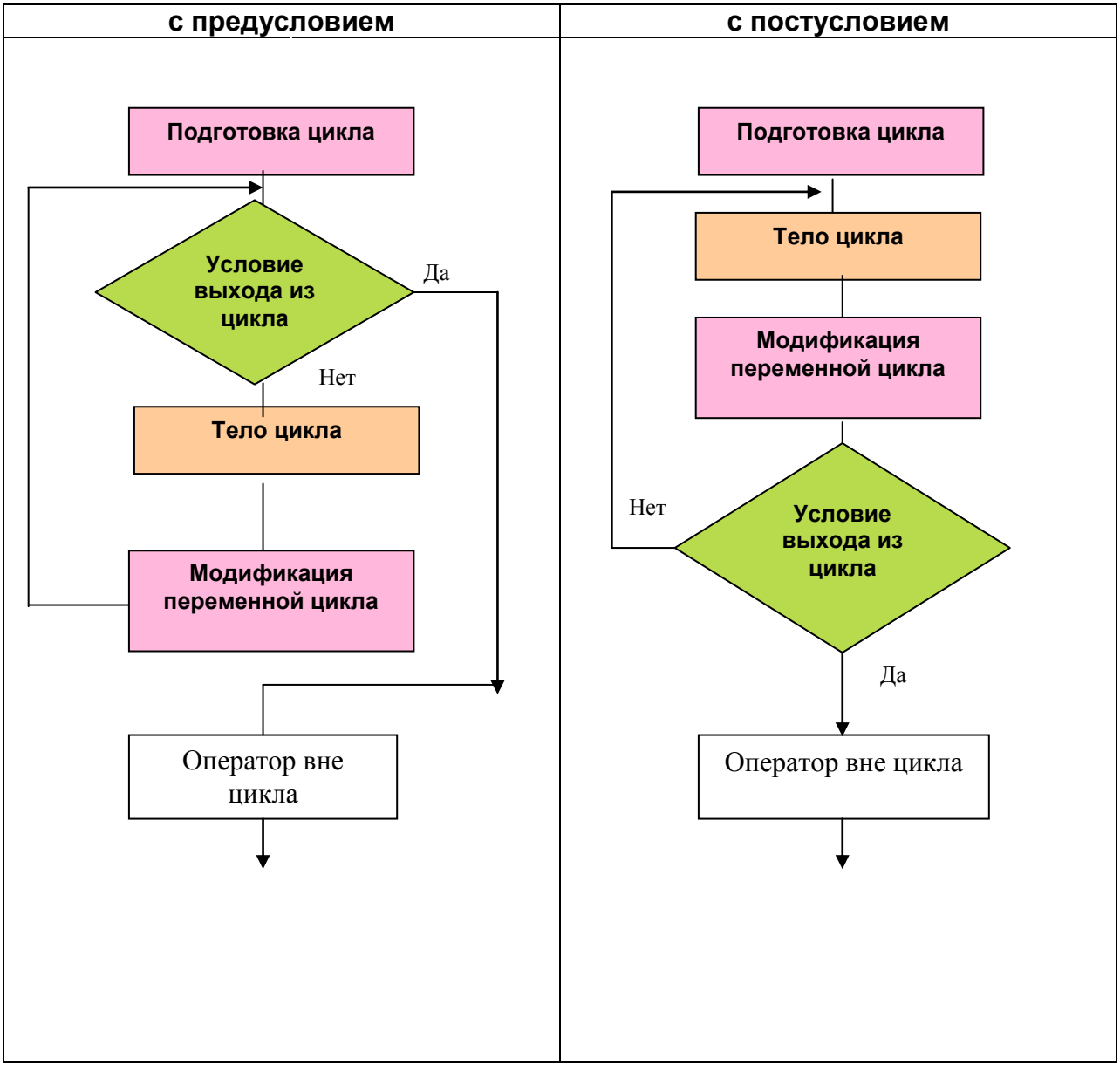
3. **Модификация переменных цикла** перед каждым новым его повторением.

4. **Управление циклом** – проверка условия продолжения (или окончания) цикла и переход на начало тела цикла, если выполняется условие продолжения цикла (или выход из цикла по его окончанию).

Базовые циклические структуры

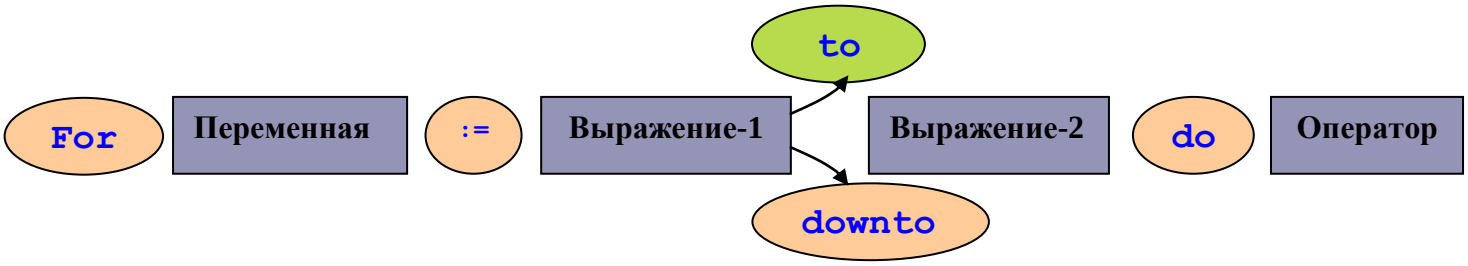


Блок - схемы циклических процессов



Оператор цикла с параметром (For)

Число повторений заранее известно



Предусматривает повторное выполнение <**оператора**> с одновременным изменением по правилу арифметической прогрессии значения, которое присваивается переменной цикла.

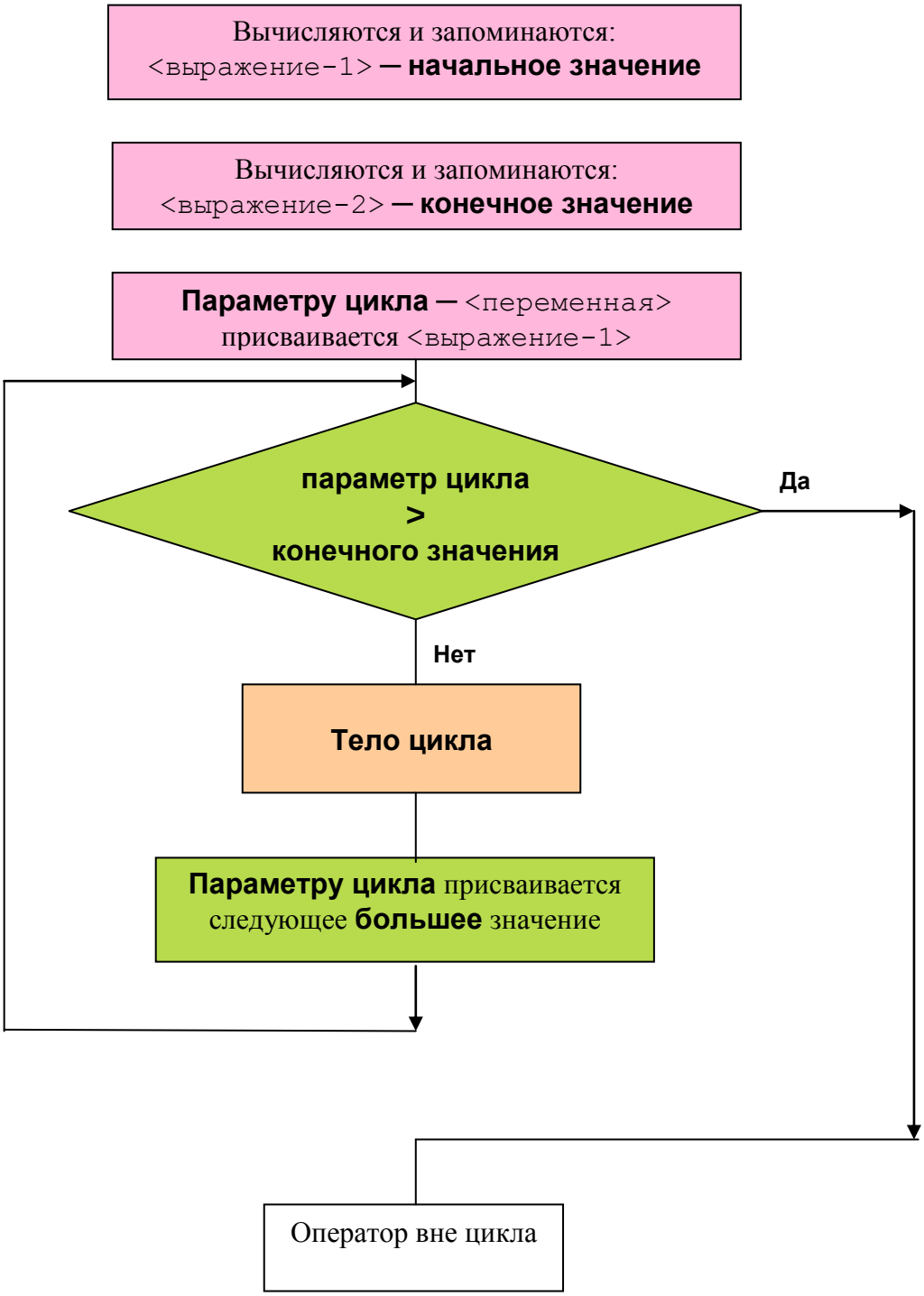
<**переменная**> - параметр цикла; является переменной порядкового типа;

<**выражение-1**> - выражение, которое определяет начальное значение параметра цикла;

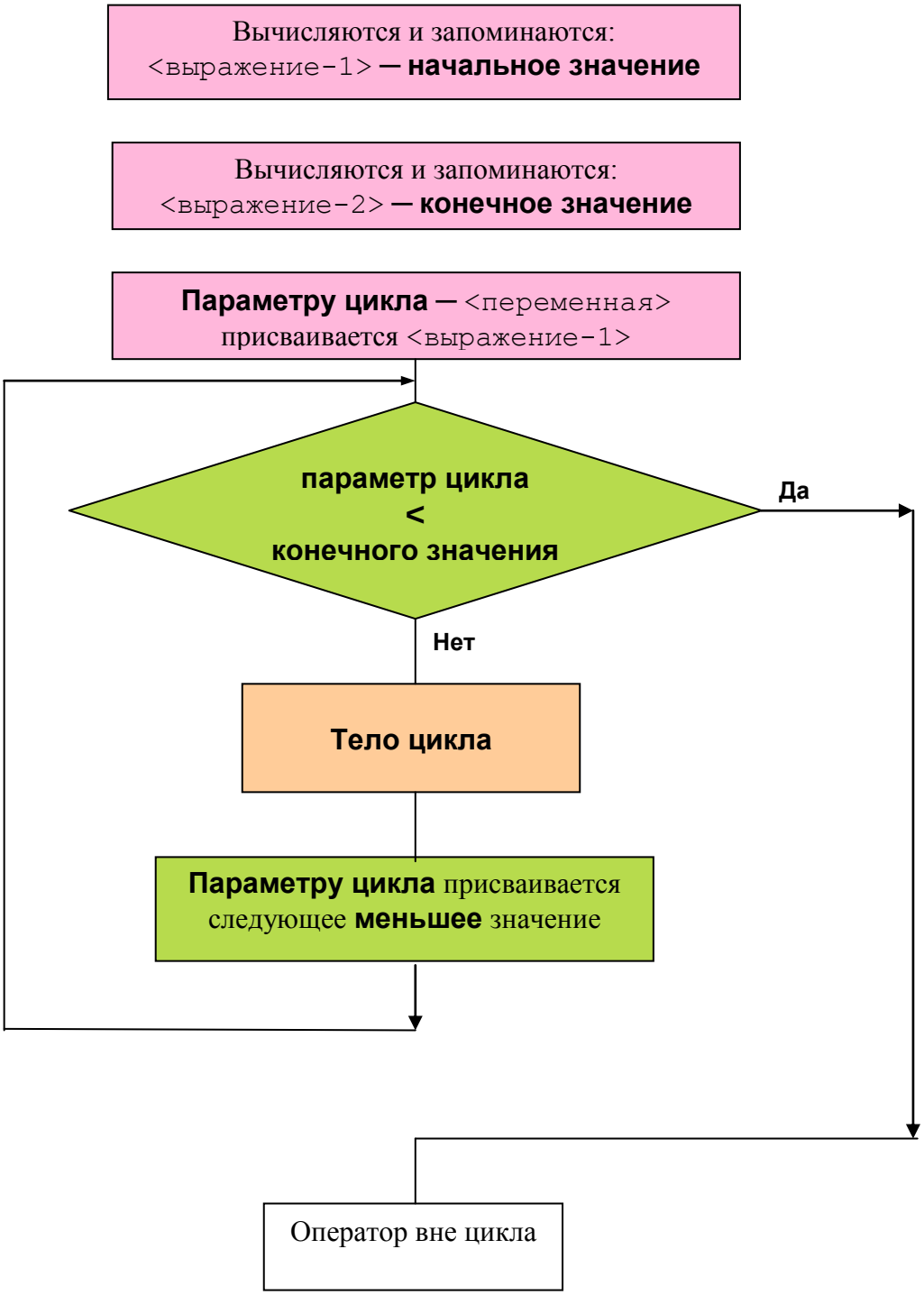
<**выражение-2**> - выражение, которое определяет конечное значение параметра цикла;

<**оператор**> - выполняемый оператор цикла.

Порядок выполнения оператора For – to



Порядок выполнения оператора For – downto



Правила организации цикла:

1. Значения параметра цикла – <**переменная**>,
начального значения – <**выражение-1**>,
конечного значения – <**выражение-2**>
должны быть одинакового порядкового типа (целый тип, символьный, логический); нельзя использовать вещественный тип.

2. Цикл не выполняется вообще, если:
начальное значение больше, чем конечное для **For – to**
начальное значение меньше, чем конечное для **For - downto**.

3. После ключевого слова **Do** может стоять **только один оператор**. Если необходимо поставить два или более оператором, то необходимо использовать составной оператор **begin**…**end**.

Запрещается

1. Изменять переменную цикла.

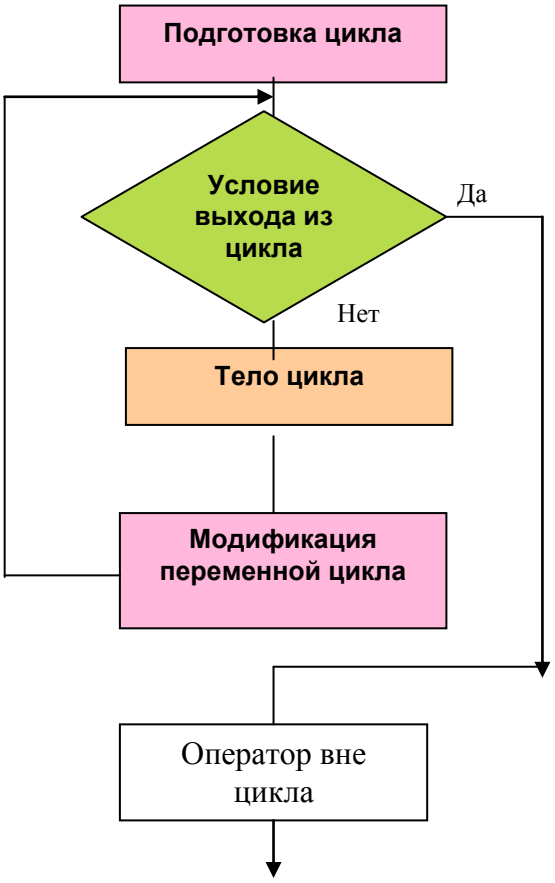
```
For I:= 1 to N do
    begin
      Y:= Y*A;
      I:= I+2
    end;
```

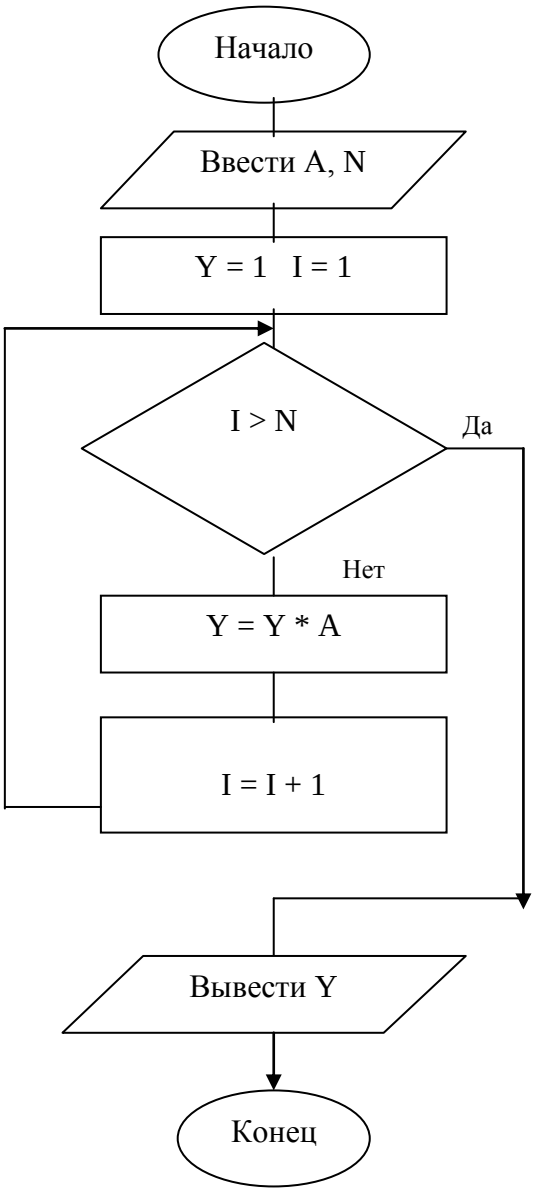
2. Входить в цикл с помощью оператора **Goto**, так как в этом случае начальное и конечное значения параметра цикла не будет определено.

```
goto METKA1;
For I := 1 to N do
    begin
 METKA1: Y := Y * A;
    end;
```

Пример

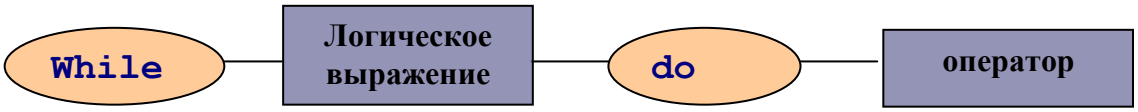
Вводятся A и N. Найти значение $Y=A^N$.





```
Program DemoFor;
Var
    A, N, I: Integer;
    N    : LongInt;
Begin
    Write( ' Значение A= ' );
    Readln(A);
    Write( ' Значение N= ' );
    Readln(N);
    Y:=1;
    For I:=1 To N do Y:=Y*a;
    Writeln('Y= ',Y );
End.
```

Оператор цикла с предусловием (While)



Предусматривает повторное выполнение <*оператора*>.

Перед каждым очередным выполнение производится проверка значения <*логического выражения*>, которое служит критерием повторения.

Если это выражение имеет значение:

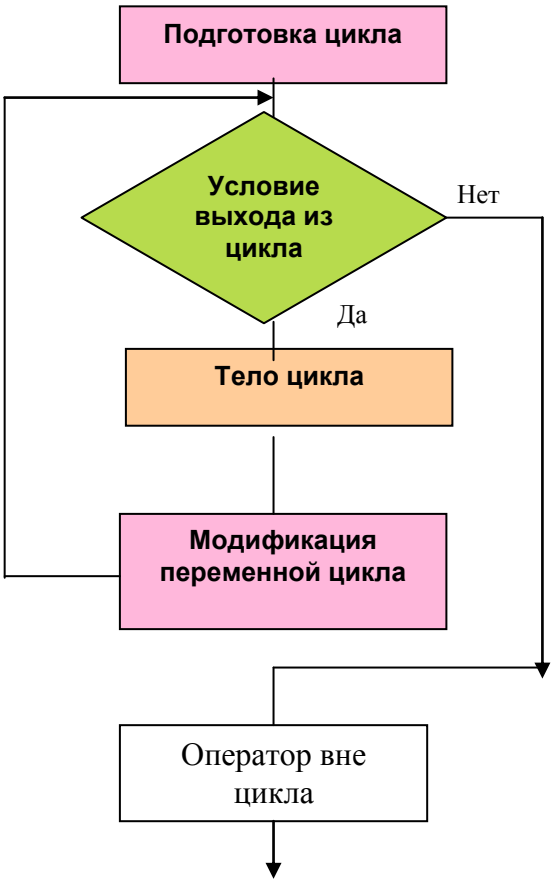
`True`, то выполняется очередная итерация;

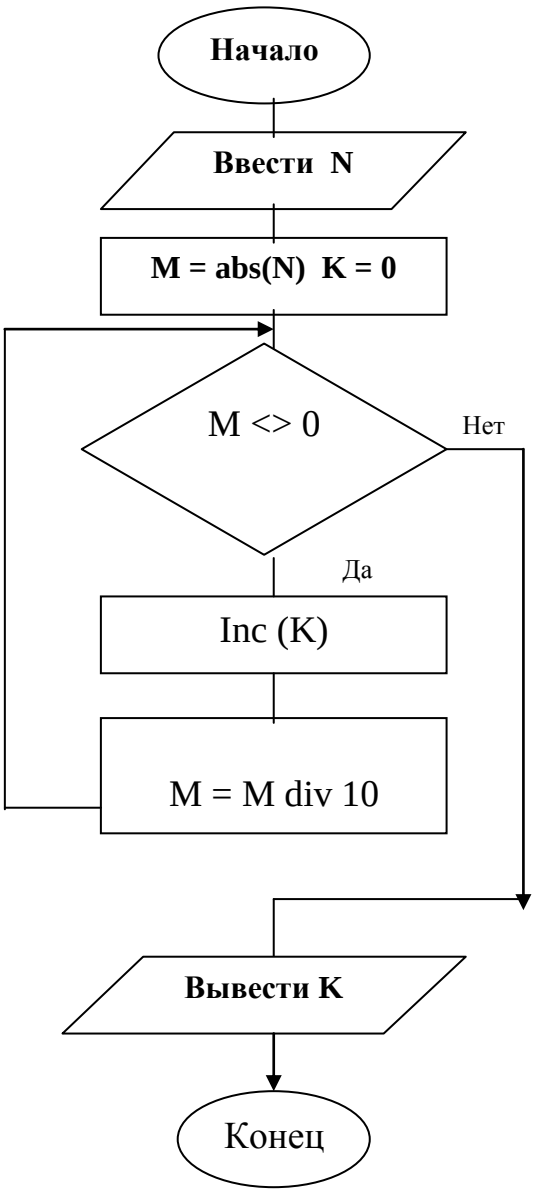
`False`, то выполнение оператора цикла заканчивается.

Если <*логическое выражение*> с самого начала имеет значение **`False`**, то цикл не выполняется ни разу.

Пример

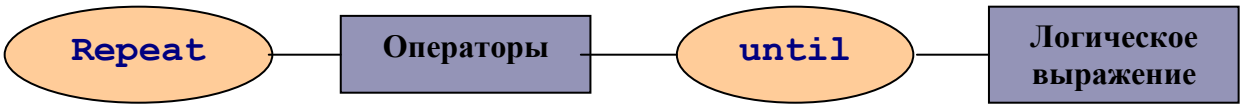
Дано число N. Подсчитать количество цифр данного числа.





```
Program DemoWhile;
Var M, N: LongInt;
    K   : Byte;
Begin
    WriteLn (' Введите целое число ');
    ReadLn (N); M:= abs(N);
    K:= 0;
    While M<>0 do
        begin
            Inc(K);
            M:= M div 10
        end;
    WriteLn('В числе ',N,' -- ',K,' цифр ')
End.
```

Оператор цикла с постусловием Repeat



Предусматривает повторное выполнение <*операторов*>.

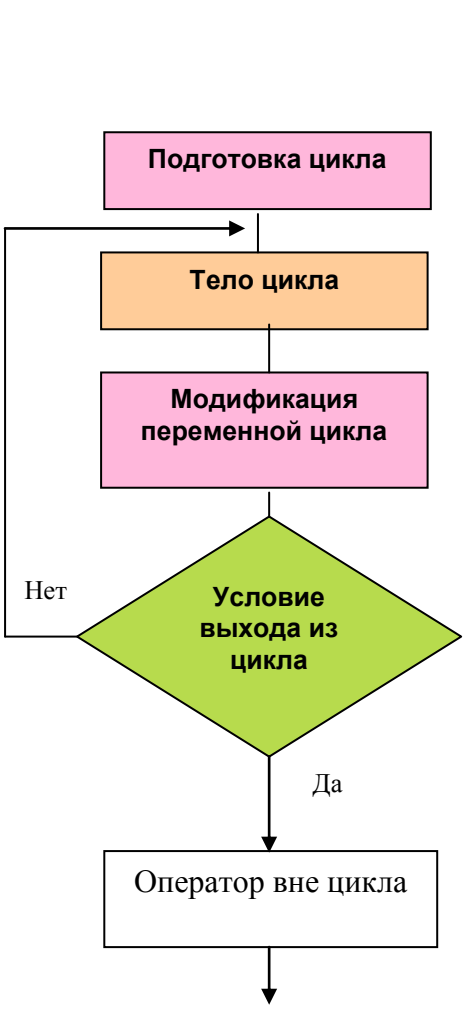
Отличается от **`While`**, тем, что условие проверяется после выполнения очередной итерации и критерием прекращения цикла является равенство <*логического выражения*> константе **`True`**.

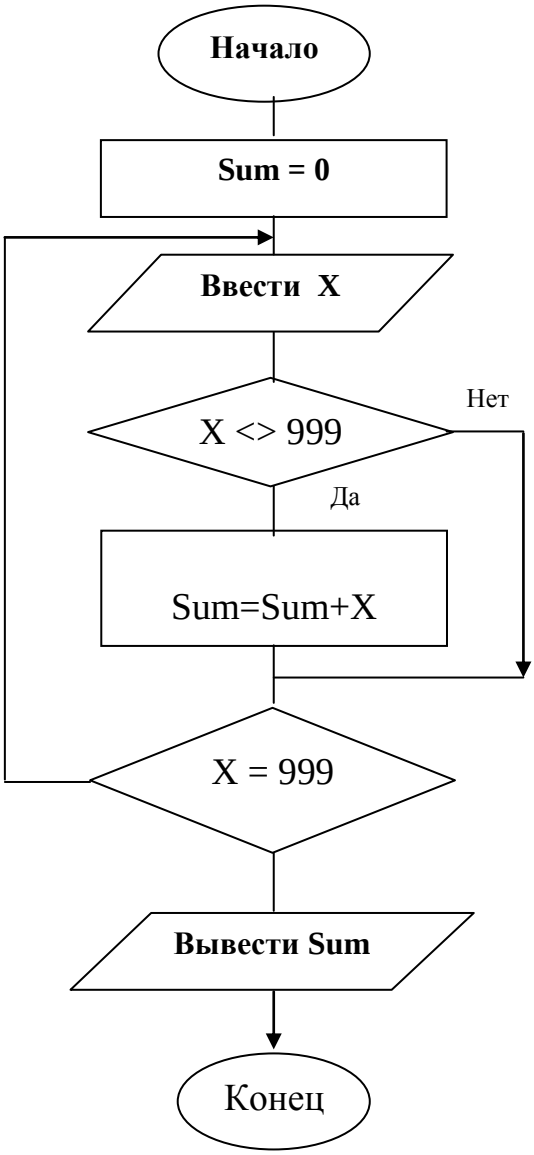
Если <*логическое выражение*> имеет значение **`False`**, то цикл повторяется.

Гарантируется хотя бы одно выполнение цикла.

Пример

Вводит и суммирует любое количество целочисленных значений. Если введено значение 999, то на экран выводится результат суммирования.





```
Program DemoRepeat;
Var
     X    : Integer;
     Sum  : Real;
Begin
     Sum := 0;
     Repeat
          Write( ' Значение X= ' );
          Readln(X);
          If X <> 999 then Sum:= Sum+X;
     until X = 999;
     Writeln('Сумма введенных чисел= ',Sum );
End.
```