

Операторы цикла

- **Цикл** — многократное повторение последовательности действий по некоторому условию.
- Известны три типа циклических алгоритмических структур: цикл с предусловием, цикл с постусловием и цикл с параметром.
- В VBA существуют операторы, реализующие все три типа циклов.

Цикл с предусловием (цикл-пока) —

наиболее универсальная циклическая структура. Он организует выполнение операторов, составляющих тело цикла, неизвестное заранее число раз. Реализуется оператором *While*. Формат оператора:

Do While <условие>

<тело цикла>

Loop

Здесь **Do**, **While**, **Loop** – зарезервированные слова;

<условие> – выражение логического типа;

<тело цикла> – операторы VBA.

Алгоритм работы оператора следующий.

Вначале вычисляется значение выражения **<условие>**.

Если **<условие>** имеет значение True, выполняется **<тело цикла>**; после чего вычисление значения выражения **<условие>** повторяется.

Если **<условие>** имеет значение False, оператор прекращает свою работу.

Таким образом, выход из цикла осуществляется, если логическое выражение принимает значение ложь.

Истинность логического выражения проверяется вначале каждого прохождения цикла, поэтому тело цикла может не выполняться ни разу.

Цикл с постусловием (цикл-до)

позволяет организовать многократное выполнение операторов, если число повторений заранее неизвестно.

Цикл с постусловием может быть записан в одном из следующих видов:

Do Until <условие>

<тело цикла>

Loop

или

Do

<тело цикла>

Loop Until <условие>

Здесь **Do**, **Until**, **Loop** – зарезервированные слова;

<условие> – выражение логического типа;

<тело цикла> – операторы VBA.

Оператор работает по следующему алгоритму.
Вначале выполняется **<тело цикла>**, после чего вычисляется значение логического выражения **<условие>**.

Если его значение есть False, операторы, образующие **<тело цикла>**, повторяются. В противном случае оператор завершает свою работу.

То есть выход из цикла осуществляется, если логическое выражение принимает значение True (истина).

Поскольку значение логического выражения вычисляется в конце каждого прохождения цикла, тело цикла выполнится хотя бы один раз.

Цикл с параметром (цикл со СЧЕТЧИКОМ, ЦИКЛ – для

служит для организации циклов с заранее известным числом повторений.

Синтаксис оператора:

For <параметр> = <начальное значение> **To** <конечное значение> [**Step** <шаг>]

<тело цикла>

Next

Здесь **For**, **To**, **Step**, **Next** – зарезервированные слова VBA;

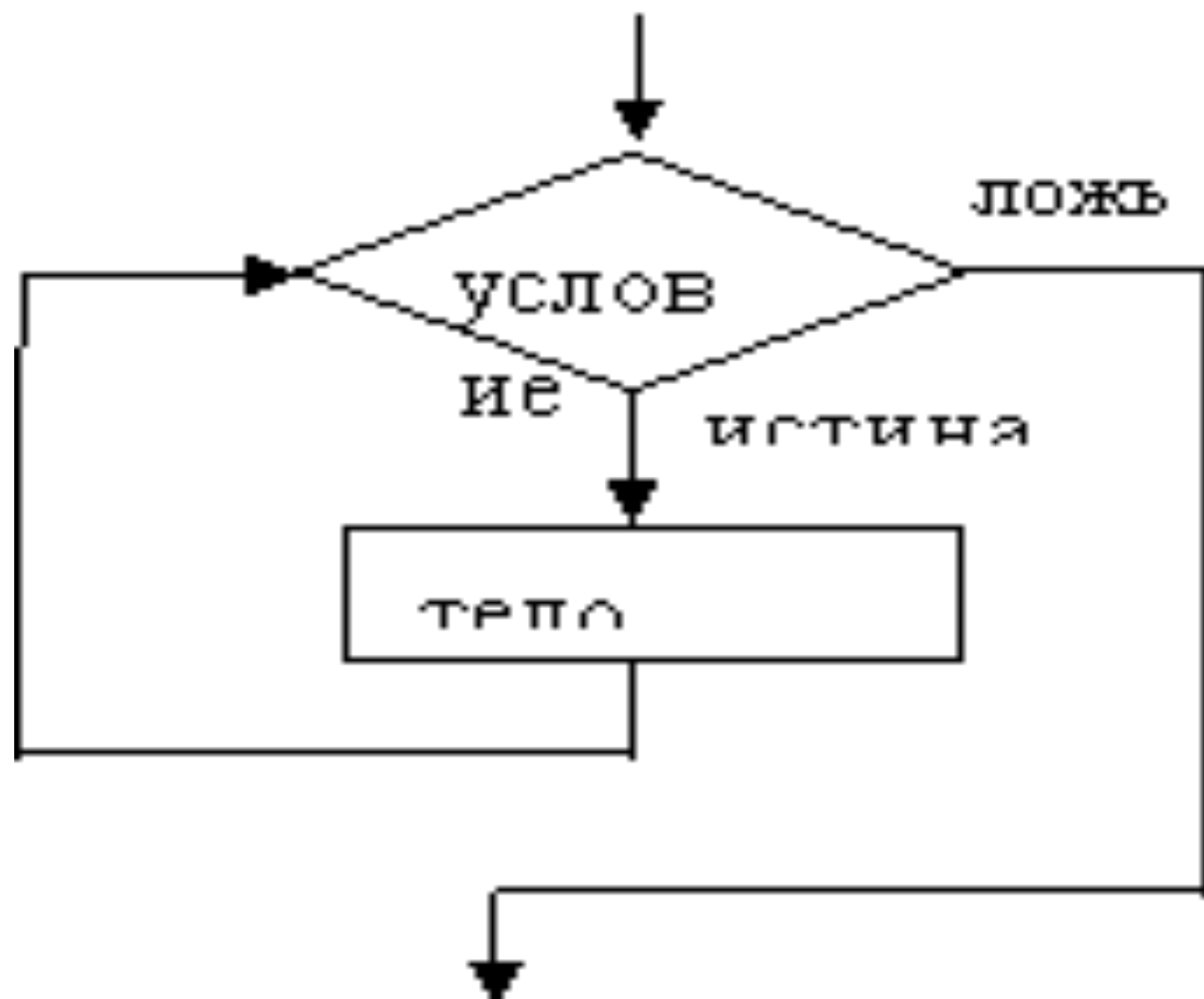
<параметр> – простая переменная порядкового типа ;

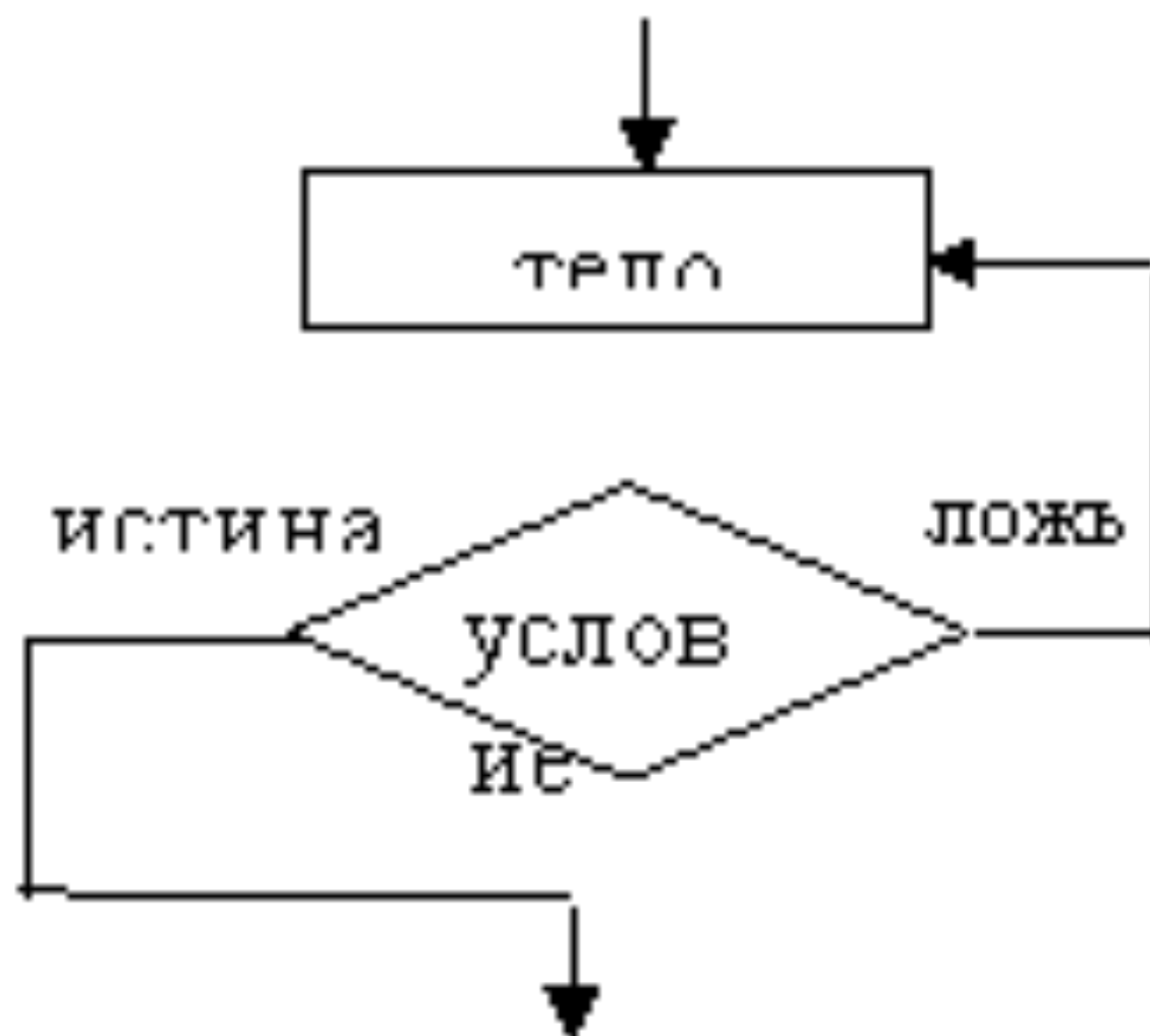
<начальное значение> – выражение того же типа что и <параметр>, определяющее начальное значение параметра;

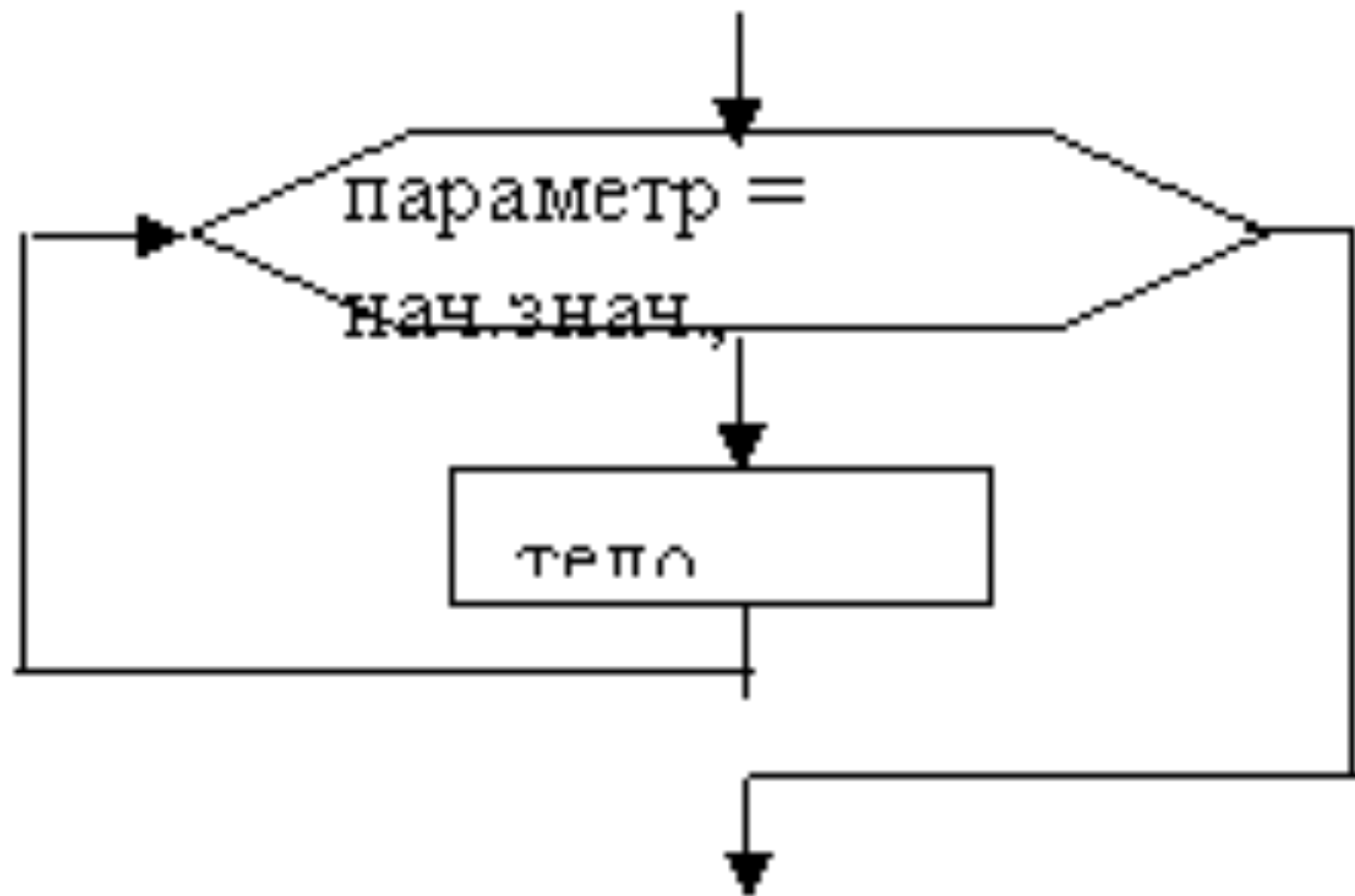
<конечное значение> – выражение того же типа, определяющее конечное значение параметра;

<шаг> – некоторое значение типа <параметр>, задающее, на сколько изменяется значение параметра при каждом проходе цикла;

<тело цикла> – операторы VBA.







Пример 1

Найти сумму десяти случайных чисел.

Напишем программу, воспользовавшись циклами различных видов.

```
Dim sum1 As Integer, sum2 As Integer, i As Integer
```

```
Randomize
```

‘решение задачи с помощью цикла с предусловием

```
i = 10
```

```
Do While i > 0
```

‘цикла выполняется, пока логическое

условие истинно

```
sum1 = sum1 + Int((10 * Rnd) + 1)
```

```
i = i - 1
```

```
Loop
```

```
MsgBox "Сумма чисел=" & sum1
```

‘решение задачи с помощью цикла с постусловием

```
i = 10
```

```
Do
```

‘цикла выполняется, пока логическое

условие ложно

```
sum2 = sum2 + Int((10 * Rnd) + 1)
```

```
i = i - 1
```

```
Loop Until i = 0
```

```
MsgBox "Сумма чисел=" & sum2
```

```
End Sub
```

Заметим, что особенностью интерпретатора VBA является то, что значения переменных числовых типов перед выполнением процедуры полагаются равными 0.

Поэтому в программе отсутствуют команды присваивания вида: `sum1 = 0` и `sum2=0`.

Пример 2

'Найти максимальное из n введенных с клавиатуры чисел.

Приведем два варианта решения задачи с использованием циклов разных видов.

Option Explicit

```
Sub Max_n_while()
```

```
Dim n As Byte, k As Single, i As Byte, Max As Single
```

```
n = Val(InputBox("Введите количество чисел"))
```

```
i = 1
```

```
Do While i <= n
```

```
k = Val(InputBox("Введите число", "Ввод чисел"))
```

```
If i = 1 Then Max = k
```

```
If k > Max Then Max = k
```

```
i = i + 1
```

```
Loop
```

```
MsgBox "Наибольшее из чисел " & Max
```

```
End Sub
```

Option Explicit

Sub Max_n_until()

Dim n As Byte, k As Single, i As Byte, Max As
Single

n = Val(InputBox("Введите количество чисел"))

i = 1

Do Until i > n

k = Val(InputBox("Введите число", "Ввод чисел"))

If i = 1 Then Max = k

If k > Max Then Max = k

i = i + 1

Loop

MsgBox "Наибольшее из чисел " & Max

End Sub

Пример 3

**Найти сумму n первых членов ряда 1, 1/2, 1/3,
... 1/n,...**

```
Option Explicit
```

```
Sub Summ_n()
```

```
Dim n As Byte, i As Byte, sum As Single
```

```
n = Val(InputBox("Введите количество членов ряда"))
```

```
For i = 1 To n
```

```
sum = sum + 1 / i
```

```
Next
```

```
MsgBox "Сумма " & sum
```

```
End Sub
```

Пример 4

Найти сумму всех четных чисел в первой десятке:

```
Option Explicit
```

```
Sub Summa ()
```

```
Dim j As Integer, sum As Integer
```

```
For j = 2 To 10 Step 2
```

```
sum = sum + j
```

```
Next
```

```
MsgBox "Сумма равна " & sum
```

```
End Sub
```



```
Public Sub Таблица()  
Dim x As Double, y As Double, a As Double, b As Double  
a = Val(InputBox("Введите начало промежутка", «Ввод a»))  
b = Val(InputBox(«Введите конец промежутка", «Ввод b»))  
H= Val(InputBox(«Введите шаг", «Ввод h»))  
x = a  
Cells(1, 1) = "x"  
Cells(1, 2) = "y"  
i = 2  
Do While x <= b + h / 2  
y = x ^ 2  
Cells(i, 1).Value = x  
Cells(i, 2) = y  
x = x + h  
i = i + 1  
Loop  
End Sub
```

```
Public Sub Prg_4()
```

```
‘FOR
```

```
Dim i As Integer
```

```
Dim f As Single
```

```
Dim g As Single
```

```
Dim x As Single
```

```
Worksheets(4).Cells(1, 1).Value = "x"
```

```
Worksheets(4).Cells(1, 2).Value = "f"
```

```
Worksheets(4).Cells(1, 3).Value = "g"
```

```
x = 0.3
```

```
For i = 2 To 8
```

```
f = (0.5 * x) * Sin(x) ^ 2
```

```
g = ((2 * x) / (4 + x ^ 2)) * Log(3 + x)
```

```
Worksheets(4).Cells(i, 1).Value = x
```

```
Worksheets(4).Cells(i, 2).Value = f
```

```
Worksheets(4).Cells(i, 3).Value = g
```

```
x = x + 0.01
```

```
Next i
```

```
End Sub
```

Условие: $g(x) = \frac{2x}{4+x^2} \ln(3+x)$

$f(x) = 0,5 \sin^2 x$

$g(x), f(x) - ?$

лр 1-4_трк...М

Гла | Вст | Раз | Фо | Даг | Рец | Вид

L30 *fx*

	A	B	C	D	E	F
1	x	f	g			
2	0,3	0,0131	0,175148			
3	0,31	0,014424	0,181174			
4	0,32	0,015832	0,187202			
5	0,33	0,017326	0,19323			
6	0,34	0,018906	0,199257			
7	0,35	0,020576	0,205281			
8	0,36	0,022337	0,211303			
9						
10						
11						
12						

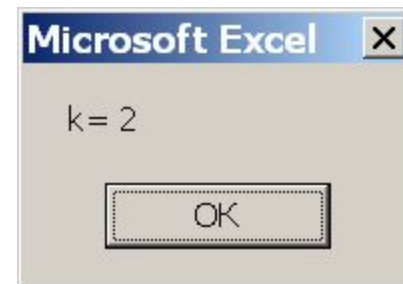
Задача 1 Лист4 За

Готово 100%

Пример.4. Элементы последовательности заданы рекуррентно по формуле $U_i = U_{i-1}/2$. Составить программу вычисления числа элементов последовательности, удовлетворяющих указанному неравенству $U > 1$, если $U_0 = 6$, а значение i изменяется от 1 до 20

- **Текст программы:**

```
Public Sub Prg_6()  
    ' Число элементов последовательности  
    Dim u As Single  
    Dim k As Integer  
    Dim n As Integer  
    u = 6  
    k = 0  
    For n = 1 To 20 Step 1  
        u = u / 2  
        If u > 1 Then  
            k = k + 1  
        End If  
    Next n  
    MsgBox ("k=" + Str(k))  
End Sub
```



Оператор **While** (пока)...**Wend** (конец цикла)

Структура оператора:

While

УСЛОВИЕ

Операторы тела цикла

Wend

Пример

Count = 0

While Count < Number

Print Count

Count = Count + 1

Wend

Задача 6. *Посчитать произведение чисел, вводимых с клавиатуры до тех пор, пока не встретится 0.*

Решение. Здесь заранее не известно, сколько чисел будет введено, поэтому лучше воспользоваться циклом While:

```
Sub Произведение ()  
Dim a As Integer, P As Integer  
a = InputBox("Введите ненулевое число")  
P = 1  
While a <> 0  
P = P * a  
a = InputBox("Введите число")  
Wend  
MsgBox (P)  
End Sub
```

Досрочный выход из цикла. При использовании циклических конструкций может возникнуть необходимость досрочного выхода из цикла. Например, получен искомый результат, а условие цикла еще истинно и позволяет продолжить исполнение этого оператора. В языке VBA оператором досрочного выхода является **Exit**, причем в циклах **For** он имеет вид **Exit For**, а в циклах, начинающихся с **Do**, он имеет вид **Exit Do**. Например, программа

```
a = 7
Do
    a = a - 1
    If a = 5 Then
        Exit Do
    End If
```

Loop Until a < 0

даст на выходе значение 5, т. к. цикл принудительно прервался, когда значение переменной a стало равным пяти. Того же результата можно достичь и в программе с циклом **For**:


```
k = 2
m = 4
For i = k + 1 To m * 2 + 1 Step 0.5
    k = k + i
    If k = 5 Then
        Exit For
    End If
Next
```