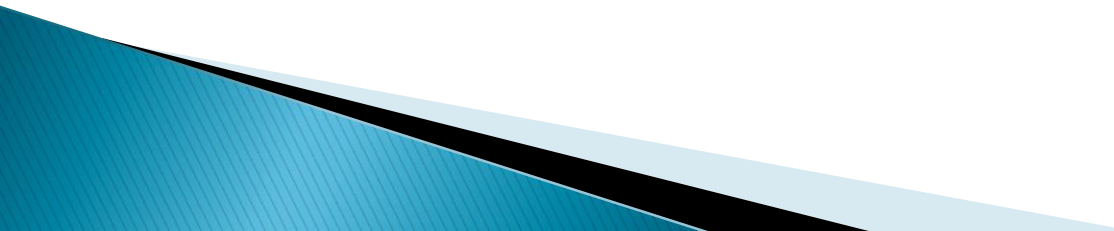


# Visual Basic for Applications (VBA)

Операторы для  
реализации циклического  
алгоритма

# Циклические алгоритмические структуры

Цикл – это такая форма организации действий, при которой одна последовательность действий повторяется несколько раз или не разу, до тех пор , пока выполняются некоторые условия.



# Существуют три вида циклов:

1. цикл «До» (с постусловием);
2. цикл «Пока» (с предусловием);
3. цикл «Для» (с параметром).

Все циклы состоят из нескольких этапов:

- ▶ Подготовка цикла, в которую входят начальные присвоения;
- ▶ Тело цикла – команды повторения цикла;
- ▶ Условие повторения – обязательная часть циклов “До” и “Пока”.

Циклы могут быть реализованы с использованием условных операторов или специальных операторов цикла

# Рассмотрим цикл “До” (с постусловием).

Это такой цикл, где тело цикла выполняется перед проверкой условия.

Его лучше использовать в той циклической структуре, когда тело цикла должно выполниться хотя бы раз.



# Цикл “Пока” (с предусловием)

- ▶ Цикл “Пока” это такой цикл, где тело цикла выполняется, пока выполняются некоторые условия . Его лучше использовать там, где сразу неизвестны начальные значения цикла.
- ▶ Этот цикл может ни разу не выполниться.

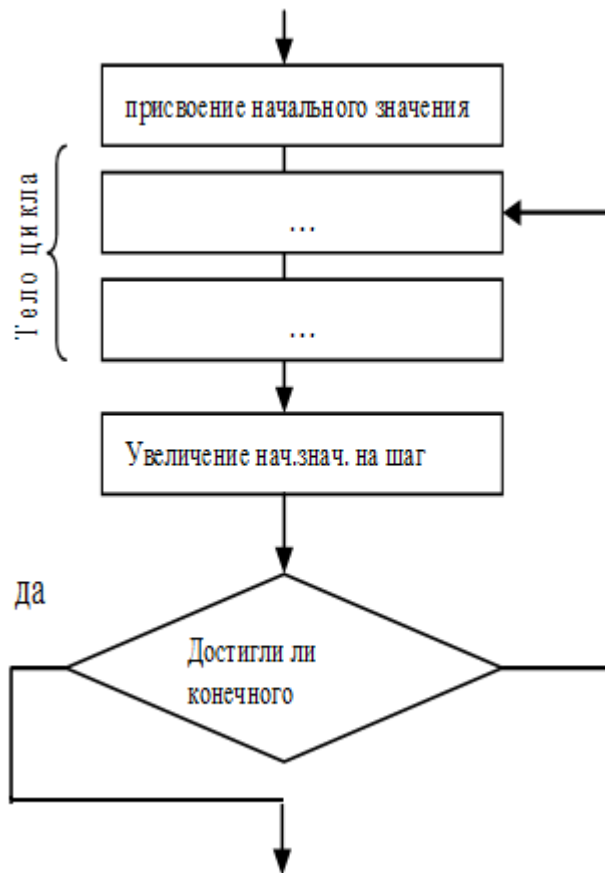


# Цикл “Для ...” или цикл с заданными параметрами

- ▶ Цикл “Для...” это цикл с параметром (счетчиком), что приводит к тому, что условие повторения не нужно.
- ▶ В этом случае обязательны два параметра: начальное и конечное значение счетчика цикла, необязательным является шаг цикла (приращение).

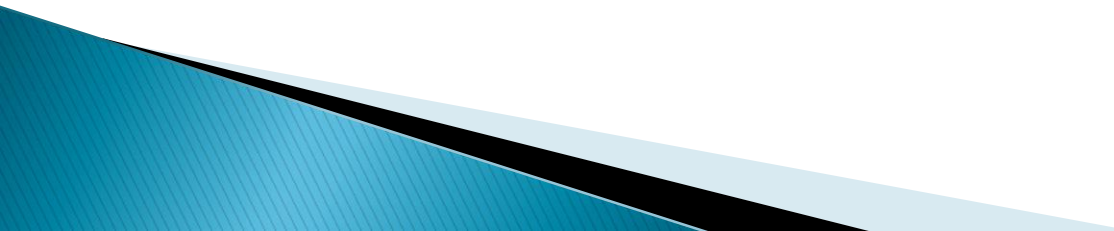
# Цикл “Для ...” или цикл с заданными параметрами.

## Блок-схема



# Операторы циклов

Для реализации циклического алгоритма можно использовать уже известный оператор условного перехода **IF**, но в VBA существуют специальные операторы для разного вида циклов.






# Оператор цикла **FOR ...NEXT** описывает цикл с параметрами

- Формат оператора:

**FOR** *счетчик* = *начало* **TO** *конец* **STEP** *шаг*

Операторы тела цикла

**NEXT** *счетчик*

- Счетчик – числовая переменная,
  - Начало – начальное значение счетчика,
  - Конец – конечное значение счетчика
  - Шаг – величина приращения счетчика.
- 

# Пример оператора **For ... Next**

...

SUM=0

FOR CH=2 TO 10 STEP 2

SUM=SUM + 2\*CH

NEXT CH


...



# Объект/элемент управления **ListBox**

Элемент управления **ListBox** отображает список элементов, в котором пользователь может выбрать один или несколько элементов.

Если все элементы не могут одновременно отобразиться в поле списка, к **ListBox** автоматически добавляется полоса прокрутки.



# Объект ListBox

## ListBox.Метод [список]

Объект характеризуется методами:

- ▶ **AddItem** выражение, индекс – добавление элементов в список, (индекс элемента считается с нуля);
- ▶ **List(Row, Column)** – возвращает элемент списка, стоящий на пересечении указанных строки и столбца;
- ▶ **Clear** – очистка поля

Примеры:

```
ListBox1.AddItem X, 1  
ListBox1.List(1, 1) = Y  
ListBox1.Clear
```

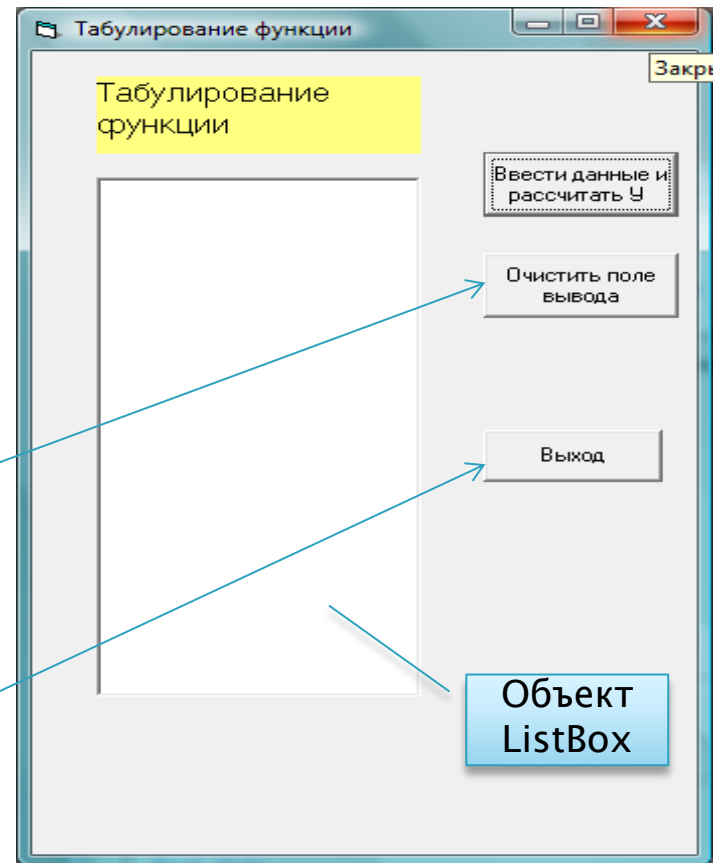
# Табулирование функции $Y(x)$ на заданном интервале $[A,B]$ с шагом $h$ :

$$Y = \begin{cases} \log \sqrt{1 + x^3} & \text{при } x > 0,5 \\ 2x - 4 & \text{при } x \leq 0,5 \end{cases}$$

Создадим форму вида:

```
Private Sub Command2_Click()  
    ListBox1.Clear  
End Sub
```

```
Private Sub Command3_Click()  
End  
End Sub
```

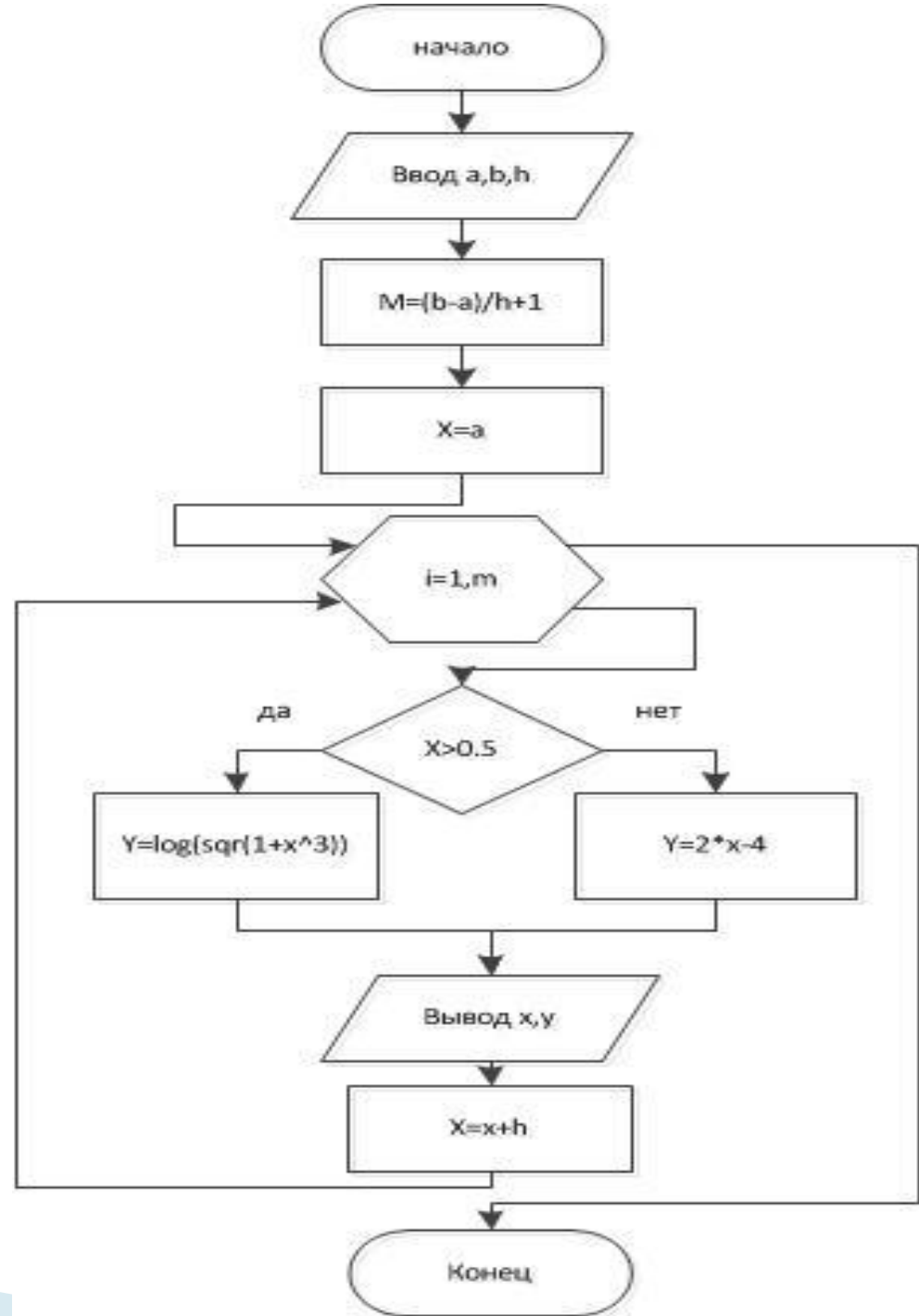


# Блок-схема алгоритма

(1 вариант)

*Условные  
обозначения:*

А – начало интервала;  
В – конец интервала;  
h – шаг изменения x;  
m – количество точек на  
интервале [A,B].



# Программный код вычисления:

```
Private Sub Command1_Click()
```

```
Dim x As Single, y As Single, a As Single
```

```
Dim b As Single, h As Single, m As Integer
```

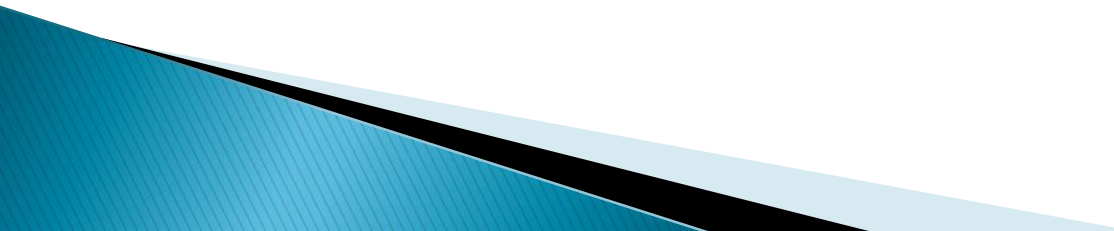
```
a = val(InputBox("Введите A", "Ввод данных", -10))
```

```
b = val(InputBox("Введите B", "Ввод данных ", 10))
```

```
h = val(InputBox("Введите H", "Ввод данных", "0.5"))
```

```
m = (b - a) / h + 1
```

```
x = a
```



## Продолжение программного кода вычисление

(1 вариант – цикл по количеству точек  $m$  и вывод в 2 колонки ListBox):

```
ListBox1.ColumnCount = 2
```

```
ListBox1.AddItem "X", 0
```

```
ListBox1.List(0, 1) = "Y"
```

```
For i = 1 To m
```

```
IF x > 0.5 Then
```

```
    y = Log(Sqr(1 + x ^ 3))
```

```
Else
```

```
    y = 2 * x - 4
```

```
End IF
```



## Продолжение программного кода вычисление (1 вариант):

```
ListBox1.AddItem x, 1
```

```
ListBox1.List(1, 1) = y
```

```
x = x + h
```

```
Next i
```

```
MsgBox ("Программа выполнена")
```

```
End Sub
```



## 2 вариант реализации алгоритма (организация цикла по X и вывод в ячейки листа excel):

**Private Sub Command1\_Click()**

Dim x As Single, y As Single, a As Single  
Dim b As Single, h As Single, m As Integer

a = InputBox("Введите A", "Ввод данных", -5)  
h = InputBox("Введите H", "Ввод данных", "0.5")  
m = InputBox("Введите M", "Ввод данных", 20)  
b = a + h \* (m - 1)

## Продолжение программного кода:

```
Cells(1, 1) = "x"
```

```
Cells(1, 2) = "y"
```

'введем индекс i – для определения  
'номера строки вывода на лист Excel

```
i = 2
```

```
For x = a To b Step h
```

```
If x > 0.5 Then
```

```
    y = Log(Sqr(1 + x ^ 3))
```

## Продолжение 2 вариант реализации алгоритма:

Else

$$y = 2 * x - 4$$

End If

Cells(i, 1) = x

Cells(i, 2) = y

i = i + 1

Next x

End sub

# Результаты (оба варианта)

Книга1 - Microsoft Excel

Файл Главная Вставка Разметка страницы Формулы Данные Рецензирование Вид Разработчик

Visual Basic Макросы Запись макроса Относительные ссылки Безопасность макросов Код Настройки Настройки COM Вставить Режим конструктора Свойства Просмотр кода Отобразить окно Источники элементов управления

D24 fx

	A	B	C	D	E	F	G	H	I	J
1	x	y								
2		0	-4							
3		0,5	-3							
4		1	0,346574							
5		1,5	0,737953							
6		2	1,098612							
7		2,5	1,405454							
8		3	1,666102							
9		3,5	1,890672							
10		4	2,087194							
11		4,5	2,261573							
12		5	2,418141							
13		5,5	2,560118							
14		6	2,689949							
15		6,5	2,809521							
16		7	2,920321							
17		7,5	3,023538							
18		8	3,120138							
19		8,5	3,210913							
20		9	3,296522							
21		9,5	3,377521							
22		10	3,454377							

Табулирование функции

табулирование на интервале [A,B] функции y

X	Y
10	3.454377
9.5	3.377521
9	3.296522
8.5	3.210913
8	3.120138
7.5	3.023538
7	2.920321
6.5	2.809521
6	2.689949
5.5	2.560118
5	2.418141
4.5	2.261573
4	2.087194
3.5	1.890672
3	1.666102
2.5	1.405454
2	1.098612
1.5	0.7379532
1	0.3465736

Ввести данные и рассчитать

Очистить список

Завершить работу

## Блок-схема алгоритма

**Задача: найти сумму и произведение первых пяти четных чисел.**

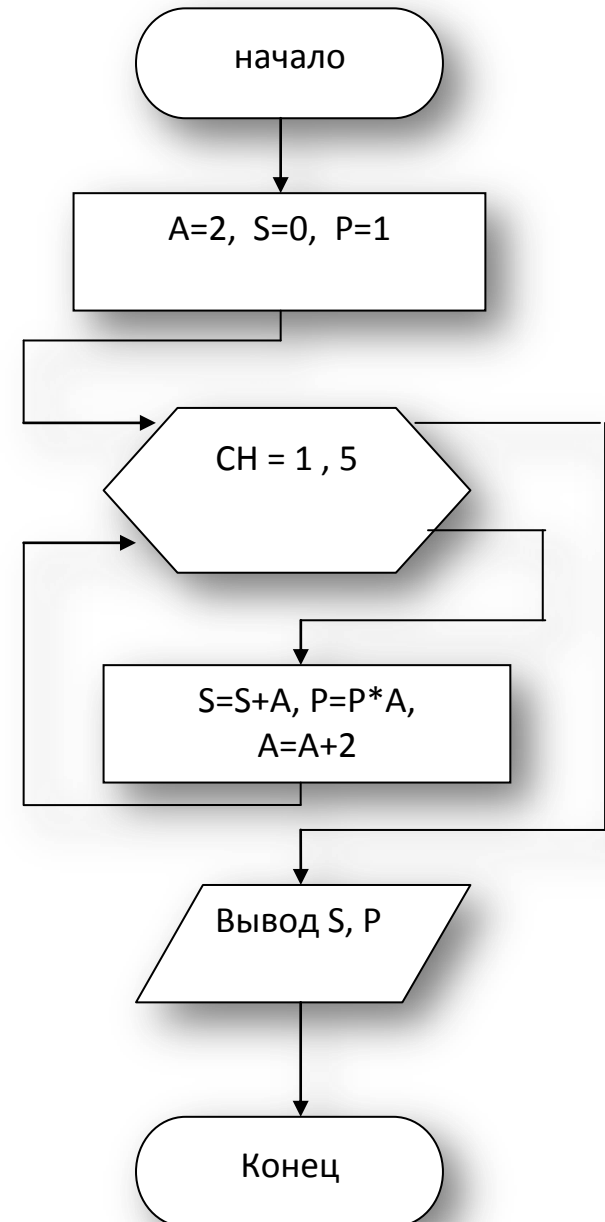
**Условные обозначения:**

**A**–число

**S**– сумма чисел

**P**– произведение чисел

**CH**– счетчик



# Программный код:

- ▶ Private Sub Command2\_Click()
- ▶ Dim a As Integer, S As Integer
- ▶ Dim ch As Integer, P As Integer
- ▶ a = 2 : S = 0 : P = 1
- ▶ For ch = 1 To 5
- ▶ S = S + a
- ▶ P = P \* a
- ▶ a = a + 2
- ▶ Next ch
- ▶ MsgBox ("Summa= " & S)
- ▶ MsgBox ("Proizvedenie = " & P)
- ▶ End Sub

# Циклы с условием

Для организации циклов с неизвестным количеством повторений используются специальные операторы:

1. **While...Wend**

2. **Do...Loop**



# Оператор **WHILE...WEND**

Выполняет серию операторов, пока указанное условие верно.

Конструкция используется тогда, когда неизвестно сколько раз должен повториться цикл.

Формат оператора:

**WHILE** условие  
Тело цикла  
**WEND**

где **условие** – числовое выражение, которое VB оценивает как истинное (не-ноль) или ложное (ноль).

# Оператор **WHILE...WEND**

## Примечание:

операторы, формирующие тело цикла, должны каким либо образом влиять на переменные, входящие в условие (т.е. изменять их значение в теле цикла), иначе произойдет «зацикливание» программы.

## *Пример:*

...

S=0 : I=1

**WHILE** S<100

S=S+3^I

I=I+1

**WEND**

Задача : проверить когда значения левой и правой части будут отличаться не более чем на 0,001.

$$\ln 2 = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} \pm \dots \pm \frac{1}{n}$$

Запишем ряд в общем виде:

$$s = \sum_{i=1}^n (-1)^{i+1} \frac{1}{i}$$

Фрагмент программы:

...

E=0.001

L=LOG(2)

S=0

I=1

**WHILE** ABS (L-S)>E

S=S+(-1)^(i+1)\*(1 /i)

i=i+1

**WEND**

MsgBox ("S=" & S & "L=" L & "i=" & i)

...

# Оператор DO...LOOP

Оператор предоставляет более удобный способ для выполнения операторов в программном цикле.

Повторяет блок операторов, пока условие верно, или до тех пор, пока условие не станет верным.

Управляющее условие может быть размещено как в начале цикла, так и конце цикла.

# Оператор **DO .... LOOP** имеет 4 формы записи:

## Цикл с предусловием

**Do While** условие  
Блок операторов  
**Loop**

Выполняется пока  
истинно условие

## Цикл с постусловием

**Do**  
Блок операторов  
**Loop While** условие

Выполняется до тех пор  
пока истинно условие

**Do Until** условие  
Блок операторов  
**Loop**

Выполняется пока  
условие не  
соблюдается

**Do**  
Блок операторов  
**Loop Until** условие

Выполняется до тех пор  
пока не соблюдается  
условие

# Оператор DO...LOOP

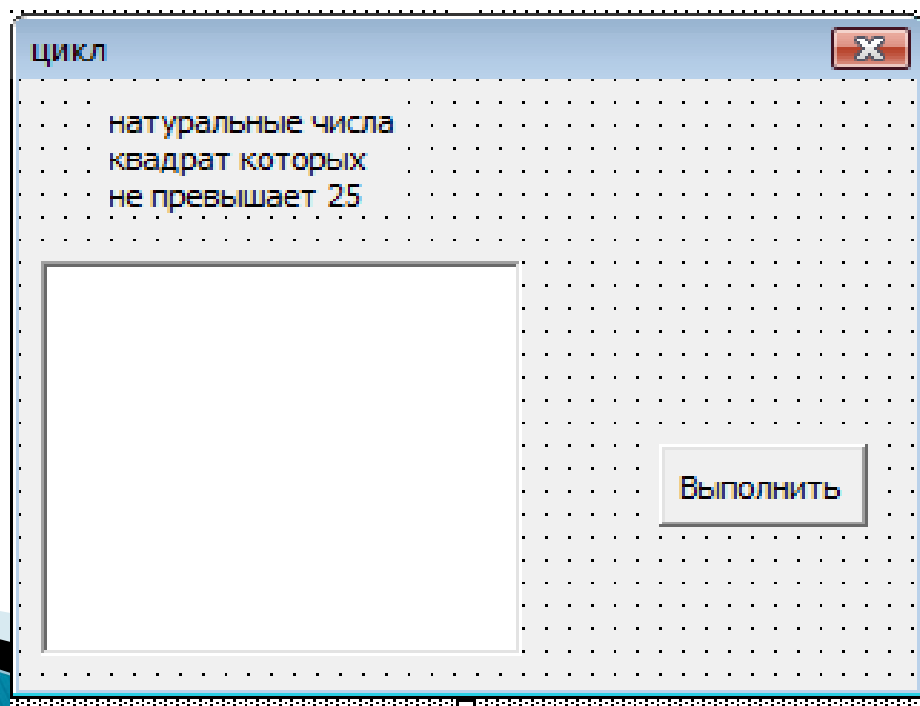
- ▶ **Условие** – числовое (логическое) выражение, которое VB оценивает как истинное (не-ноль) или ложное (ноль).
- ▶ **WHILE** – цикл выполняется, пока условие принимает значение ИСТИНА.
- ▶ **UNTIL**– цикл выполняется, пока условие принимает значение ЛОЖЬ.

Ключевое слово	«истина»	«ложь»
WHILE	Процесс продолжается	Процесс прекращается
UNTIL	Процесс прекращается	Процесс продолжается

Задача: Вывести на экран первые натуральные числа, квадрат которых не более 25.

Используем для организации цикла оператор Do...Loop с проверкой условия вверху – истина.

Создадим форму данного вида, используя для вывода элемент управления ListBox:



# Фрагмент программного кода:

...

```
Dim n As Integer
```

```
n = 1
```

```
ListBox1.ColumnCount = 2
```

```
ListBox1.AddItem "число", 0
```

```
ListBox1.List(0, 1) = "его квадрат"
```

```
Do While n ^ 2 <= 25
```

```
ListBox1.AddItem n, n
```

```
ListBox1.List(n, 1) = n ^ 2
```

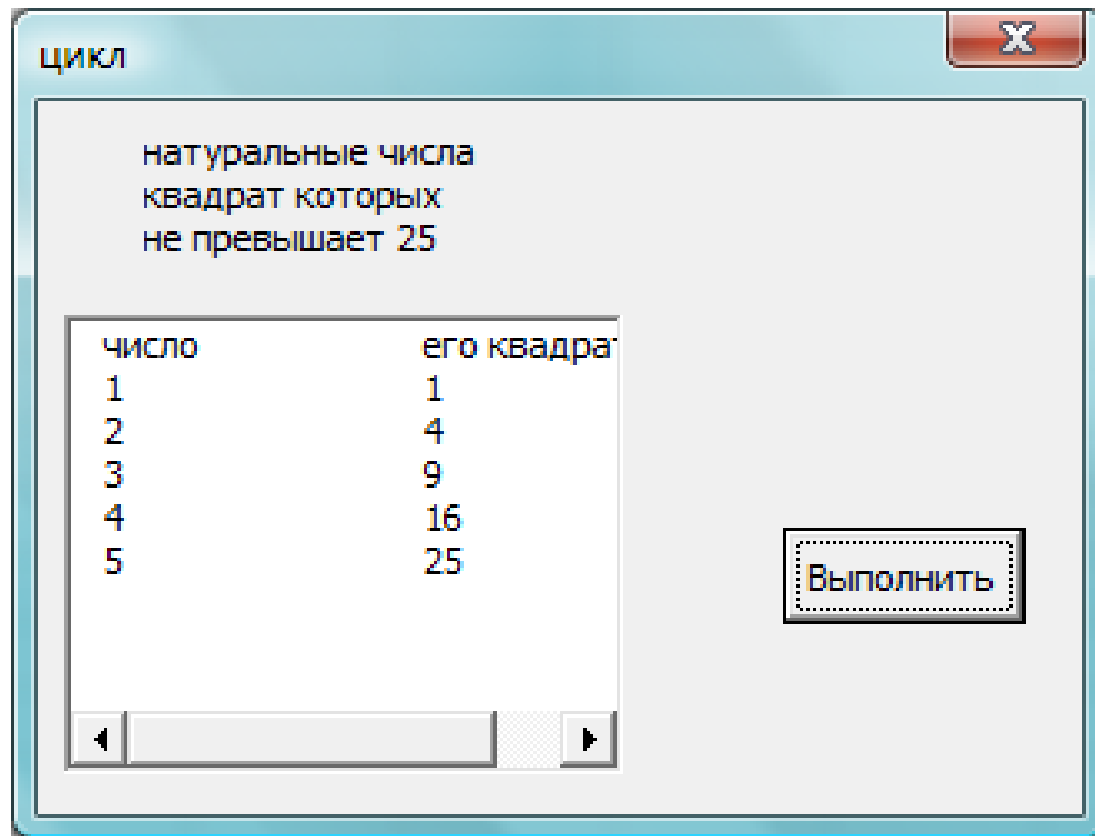
```
n = n + 1
```

```
Loop
```

...



# Результат работы:



## Другой вариант решения: Do ... Loop с проверкой сверху – ложь

N=1

ListBox1.ColumnCount = 2

ListBox1.AddItem "число", 0

ListBox1.List(0, 1) = "его квадрат"

**DO UNTIL**  $N^2 > 25$

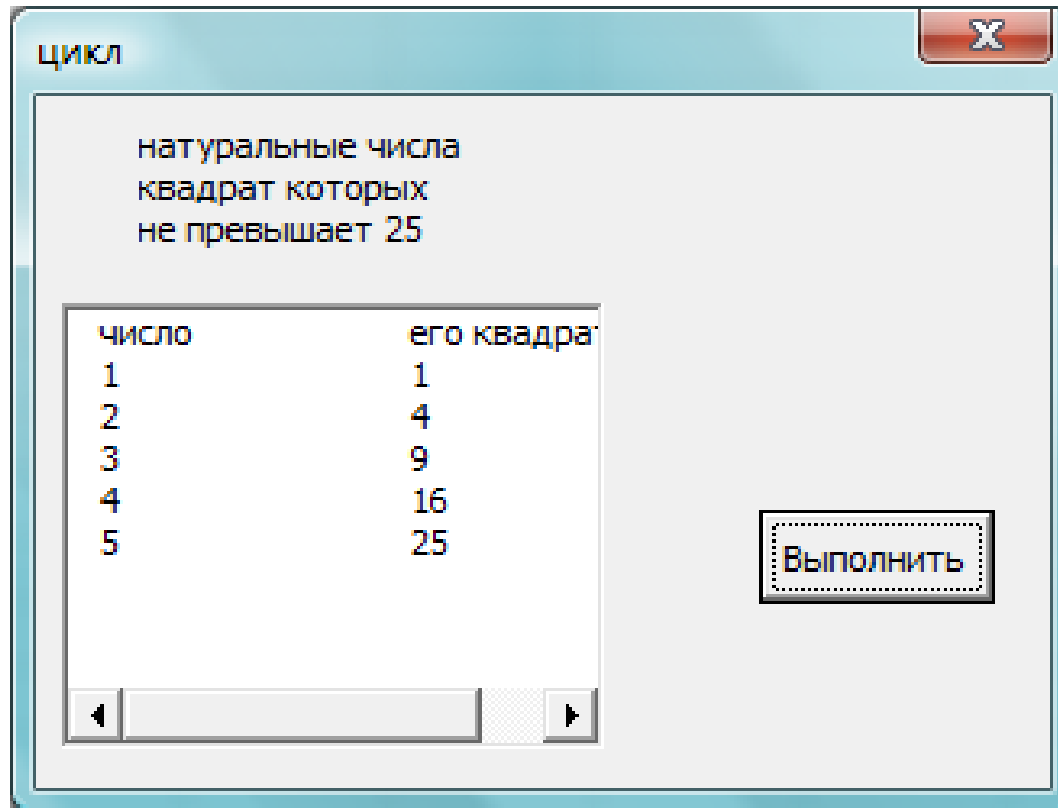
ListBox1.AddItem n, n

ListBox1.List(n, 1) =  $n^2$

N=N+1

LOOP

# Результат работы:



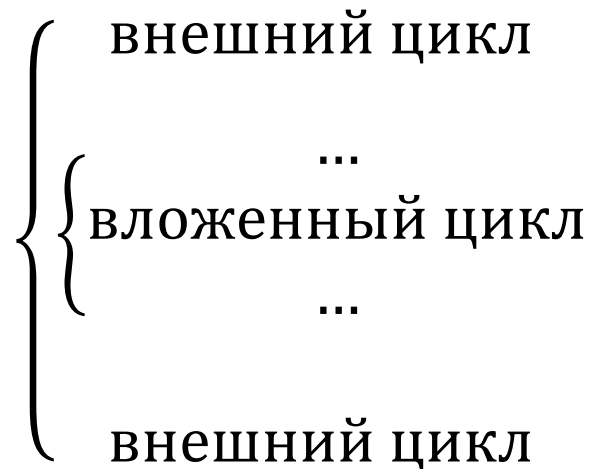
# Вложенные циклы

- ▶ В теле любого оператора цикла могут находиться другие операторы цикла.
- ▶ При этом цикл, содержащий в себе другой, называют *внешним*, а цикл, находящийся в теле первого — *внутренним (вложенным)*.
- ▶ Правила организации внешнего и внутреннего циклов такие же, как и для простого цикла.

# Вложенные циклы

- ▶ При программировании вложенных циклов необходимо соблюдать следующее условие:  
*все операторы внутреннего цикла должны полностью располагаться в теле внешнего цикла.*

Пересекать  
циклы  
нельзя

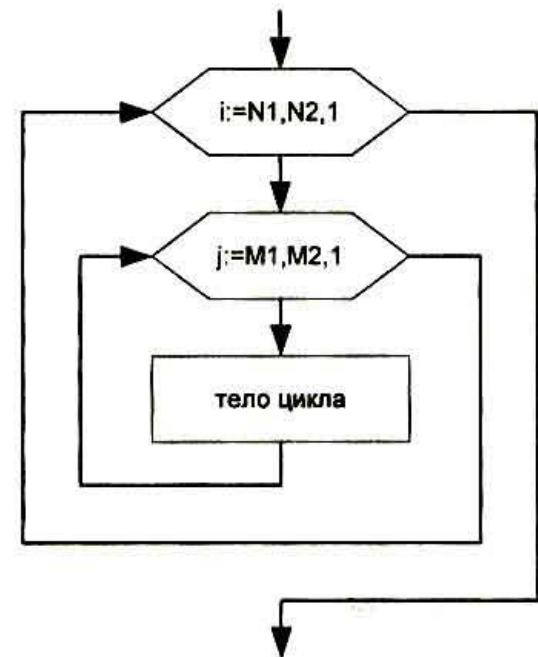


# Рассмотрим задачу вывода на экран таблицы умножения

Решение данной задачи предполагает применение вложенных циклов.

Фрагмент блок-схемы:

где  $i$  – цифра в строке,  
 $j$  – цифра в столбце.

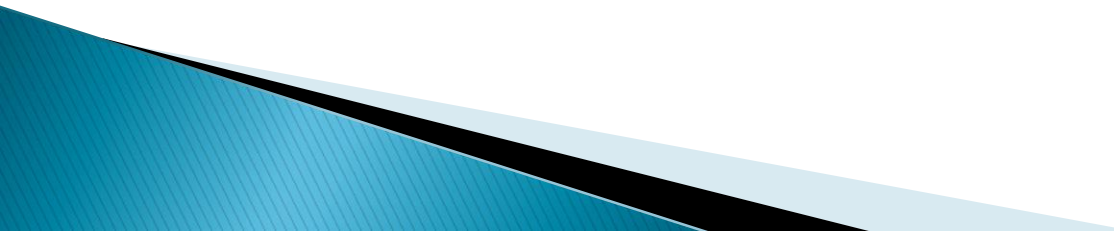


# Программный код с выводом в поле ListBox:

```
Private Sub Command1_Click()  
Dim l As Integer, j As Integer  
Dim k As Integer, kk As String  
For i = 1 To 10  
    kk = ""  
    For j = 1 To 10
```

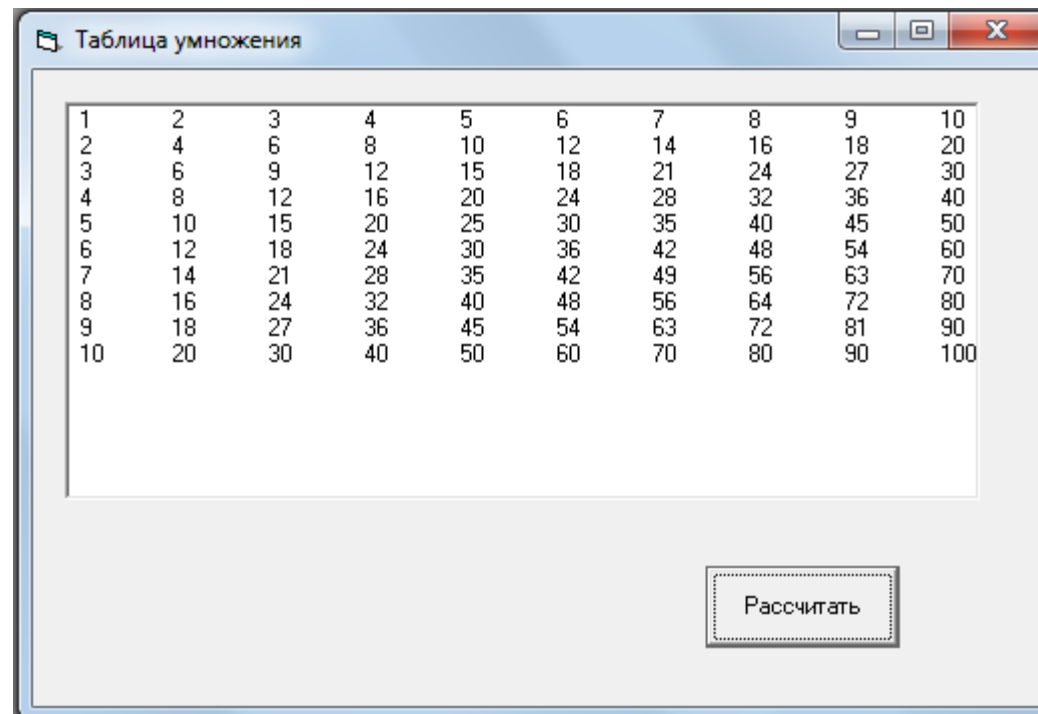
# Продолжение

```
k = i * j  
kk = kk + Str(k) + vbTab  
Next j  
ListBox1.AddItem kk  
Next i  
End Sub
```



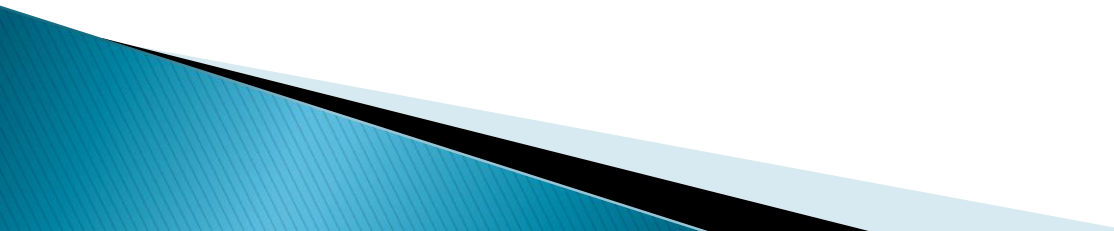


# Результат работы вывода в поле ListBox.



# Подпрограмма вывода в ячейки листа рабочей книги Excel:

```
Private Sub CommandButton1_Click()  
    Dim i As Integer, j As Integer  
    For i = 1 To 10  
        For j = 1 To 10  
            Cells(i, j) = i * j  
        Next j  
    Next i  
End Sub
```



# Результат работы:

A1		fx		1						
	A	B	C	D	E	F	G	H	I	J
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8						20
3	3	6	9	12						30
4	4	8	12	16						40
5	5	10	15	20						50
6	6	12	18	24						60
7	7	14	21	28						70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Вложенные циклы

таблица умножения на лист рабочей книги

*Составить программу для  
вычисления:*

$$A = \prod_{k=1}^3 k + \sum_{f=1}^k \sum_{g=1}^f \frac{\lg F - e^k}{g^3}$$

*Введем обозначения:*

- ▶ *внутренняя сумма **b**,*
- ▶ *средняя сумма **c**,*
- ▶ *внешнее произведение **a**.*

# Программный код:

```
Private Sub Command2_Click()  
Dim c As Single, b As Single, a As Single  
Dim k as Integer, f as Integer, g as Integer  
a = 1: b = 0: c = 0  
For k= 1 To 3  
For f = 1 To k  
For g = 1 To f  
b = b+(Log(f)/log(10)- Exp(k)) / g ^ 3
```

# Продолжение программного кода:

Next g

c = c + b

Next f

a = a \*(k+ c)

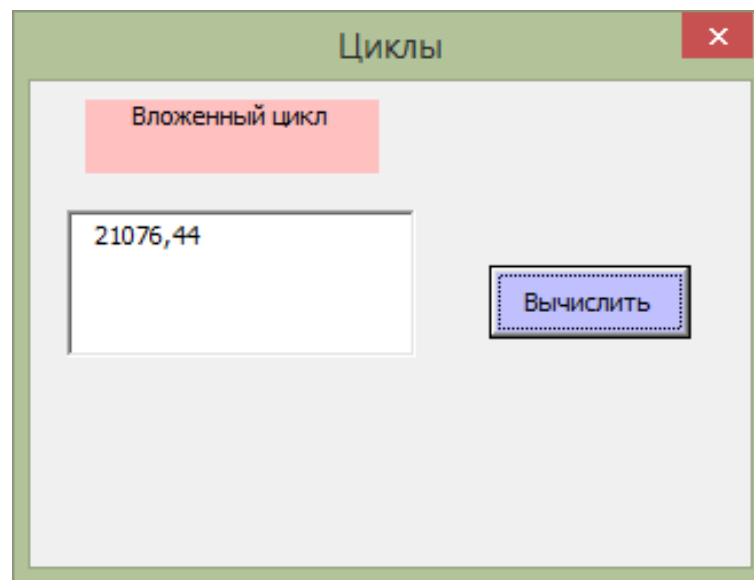
Next k

TextBox1.Text = a

End Sub



# Форма с результатом



The image shows a screenshot of a software window titled "Циклы" (Cycles). The window has a green title bar with a red close button (X) in the top right corner. Inside the window, there is a light gray background. At the top left, there is a red rectangular label with the text "Вложенный цикл" (Nested cycle). Below this label, on the left, is a white rectangular text box containing the number "21076,44". To the right of this text box is a blue rectangular button with a dashed border and the text "Вычислить" (Calculate).

# Спасибо за внимание

- ▶ ТОГУ, кафедра информатика
- ▶ Сергеева Лариса Анатольевна