ЛАБОРАТОРНАЯ РАБОТА

ПРОГРАММИРОВАНИЕ ЦИКЛИЧЕСКИХ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ

Тема - программирование циклических вычислительных процессов с использованием операторов, реализующих структуры цикла.

Цель работы - освоить на практике применение циклических алгоритмов, используя для этого различные операторы цикла, научиться отлаживать программы.

1.1 ОБЩИЕ СВЕДЕНИЯ

Циклические алгоритмы используются, когда для получения результата некоторую последовательность действий необходимо выполнить несколько раз.

Например, суммируется набор из двадцати чисел, вводимых с клавиатуры. Действия: «ввести число» и «добавить его к сумме» будут повторяться двадцать раз.

В задаче: складывать вводимые с клавиатуры числа, до тех пор, пока не встретится 0, эти действия будут повторяться до тех пор, пока будет выполняться условие «введенное число не равно нулю».

Для обозначения повторений в записи алгоритмов используют конструкции, называемые циклами. При этом могут быть следующие варианты: цикл с параметром (счетчиком), цикл с предусловием и цикл с постусловием.

Повторяющаяся группа действий называется телом цикла, однократное выполнение этой группы – шагом (или итерацией) цикла. Часть конструкции, в которой определяется, продолжать выполнение цикла или нет, называют заголовком цикла. Заголовок, как правило, содержит условие, истинность или ложность которого требует повторения операторов тела цикла.

Цикл с предусловием (рисунок 1.1).

В заголовке такого цикла содержится условие. При выполнении этого условия (результат проверки есть Истина) операторы, составляющие тело цикла, будут выполняться, в

противном случае выполняется инструкция, записанная после конца цикла.

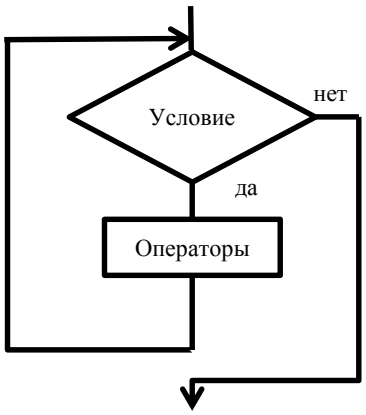


Рисунок 1.1

После достижения оператора конца цикла опять проверяется условие продолжения, и, если оно истинно, операторы тела цикла выполняются еще раз. Таким образом, чтобы цикл закончился, в теле цикла должны быть выполнены операции, в результате которых условие продолжения цикла примет значение Ложь.

В VBA имеется два варианта цикла с предусловием: **While**…**Wend** и **Do** …**Loop.**

Первый вариант:

While *Условие*
Операторы
Wend

Выход из такого цикла возможен только при невыполнении условия после слова While. Прервать выполнение цикла изнутри

нельзя. Если изначально *Условие* имеет значение False (Ложь), *Операторы* цикла не будут выполнены ни разу.

Пример

```
'Вычисление 10!
Dim N as Byte, Factor as Long
N=1
Factor=1
While N<=10
   Factor=Factor*N
   N=N+1
Wend
MsgBox Str(N) & "!=" & Factor
```

В данном примере переменным N и Factor сначала присваиваются значения 1. При входе в цикл, следовательно, проверяется условие 1<=10, результатом проверки которого является True (Истина), и выполнение цикла начинается. Внутри цикла текущее значение переменной Factor умножается на текущее значение переменной N и результат заменяет хранившееся ранее в переменной Factor значение. Затем переменная N увеличивается на 1 и опять проверяется условие продолжения цикла (2<=10). Поскольку условие истинно, снова выполняются действия внутри цикла.

После того, как в теле цикла переменная N примет значение 11 (предварительно в переменной Factor будет вычислено значение произведения 1·2·3·4·5·6·7·8·9·10), условие продолжения (11<=10) окажется ложным и цикл завершится.

Второй вариант:

Do [**While** *условие1*]
    Операторы 1
    [*Условие2* **Exit Do**]
    [*Операторы 2*]
Loop

Выход из такого цикла возможен как при невыполнении условия в начале цикла, так и изнутри тела цикла (оператор Exit Do, которому, как правило, предшествует проверка некого условия).

В квадратных скобках показаны необязательные элементы конструкции. Сами квадратные скобки вводить не нужно. Это просто означает, что то, что приведено в квадратных скобках, может отсутствовать.

Если не задано ни условие при входе в цикл (то есть условие1 подразумевается всегда истинным), ни оператор выхода из цикла Exit Do, получится «вечный» цикл («зацикливание»). Программа в таком случае сможет прерваться при возникновении какой-либо ошибки внутри цикла (например, выход за границы допустимых значений переменной), либо ее придется прерывать средствами операционной системы. При составлении программы следует стараться избегать таких ситуаций.

Пример

```
'Попытка вычисления значения 100!
Dim N as Byte, Factor as Long
N=1
Factor=1
Do  While N<100
   Factor=Factor*N
   If Factor>100000000 Then
         MsgBox "Число слишком большое"
         Exit Do
   EndIf
   N=N+1
Loop
MsgBox Str(N) & "!=" & Factor
```

Значение факториала растет очень быстро, а максимальное число, которое можно сохранить в переменной типа Long, равно примерно $2 \cdot 10^9$. Поэтому в данном примере внутри цикла каждый раз после вычисления очередного значения факториала полученное значение сравнивается с числом 10^8, и, если значение окажется

больше, вычисления прекращаются (происходит выход из цикла) и выводится результат.

Цикл с постусловием (рисунок 1.2).

Такой цикл выполнится хотя бы раз. Условие в конце – это условие выхода из цикла. То есть, для того, чтобы цикл возобновился, результатом проверки условия должно быть значение Ложь.

Синтаксис:

Do
Операторы
Loop Until *Условие*

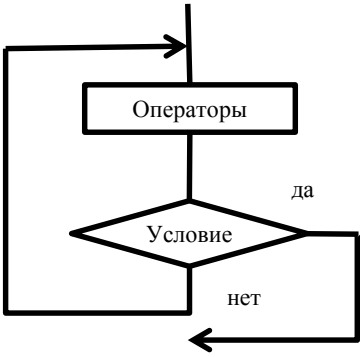


Рисунок 1.2

Пример

```
'Вычисление 10!
Dim N as Byte, Factor as Long
N=1
Factor=1
Do
   Factor=Factor*N
   N=N+1
```

Loop Until N>10
MsgBox Str(N) & "!=" & Factor

ПРИМЕР ВЫПОЛНЕНИЯ РАБОТЫ

Задача 1.1. Составить алгоритм в виде блок-схемы и программу для вычисления членов ряда, значение которых не более 32767. Члены ряда вычисляются в соответствии с выражением:

$N_{k+1}=N_k+ N_{k-1}$, при этом $N_0= N_1=1$. Результат вывести на лист MS Excel.

Решение

Составим блок-схему алгоритма (рисунок 1.3).

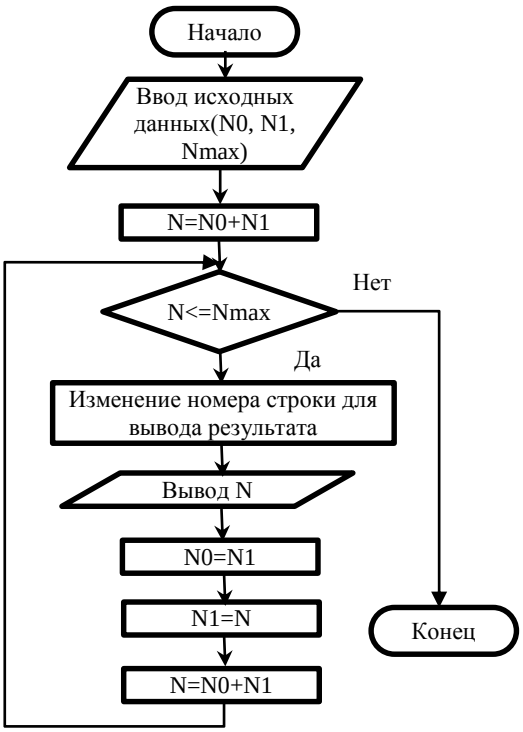
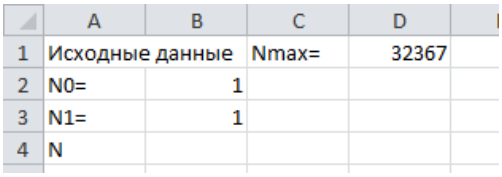


Рисунок 1.3

Лист с исходными данными приведен на рисунке 1.4.



	A	B	C	D
1	Исходные данные		Nmax=	32367
2	N0=	1		
3	N1=	1		
4	N			

Рисунок 1.4

Таким образом, значения N0, N1 и Nmax программа должна будет прочитать из ячеек B2, B3 и D1 соответственно, а вычисленные значения ряда будем выводить в столбец, начиная с ячейки B4. Для изменения номера строки при выводе результатов будем использовать переменную I, хранящую номер последнего известного члена ряда.

Текст программы на VBA приведен ниже.

```
Option Explicit
Public Sub Primer1()
Dim N0 As Long, N1 As Long, Nmax As Integer, N As Long
Dim I As Integer 'номер последнего известного члена ряда
I = 1
N0 = Range("B2")
N1 = Range("B3")
Nmax = Range("D1")
N = N0 + N1
While N <= Nmax
   I = I + 1
   Cells(I + 2, 2) = N
   N0 = N1
   N1 = N
   N = N0 + N1
Wend
End Sub
```

Часть листа с результатами работы программы показана на рисунке 1.5.

	A	B	C	D
1	Исходные данные		Nmax=	32767
2	N0=	1		
3	N1=	1		
4	N	2		
5		3		
6		5		
7		8		
8		13		
9		21		
10		34		

Рисунок 1.5

Количество операторов в тексте программы можно уменьшить на один, если использовать цикл с постусловием. Пример программы приведен ниже.

```
Public Sub Primer1_1()
Dim N0 As Long, N1 As Long, Nmax As Integer, N As Long
Dim I As Integer 'номер последнего известного члена ряда
I = 1
N0 = Range("B2")
N1 = Range("B3")
Nmax = Range("D1")
Do
   N = N0 + N1
   I = I + 1
   Cells(I + 2, 2) = N
   N0 = N1
   N1 = N
Loop Until N0 + N1 > Nmax
End Sub
```

Задача 1.2. Составить алгоритм в виде блок-схемы и программу для вычисления суммы ряда $\sum_{n=1}^{\infty}\frac{(-1)^{n-1}x^{2n-1}}{(2n-1)!}$ при предварительно введенных нескольких значениях x. Процесс вычисления каждой суммы прекратить, когда очередное слагаемое по абсолютной величине окажется менее 0,0005. Сравнить полученные суммы со значением sin(x). Исходные данные (значения x и минимальное слагаемое) получить из ячеек на листе MS Excel, результаты вычислений (значения сумм при соответствующих значениях x и sin(x)), округленные до тысячных, вывести на лист. Проверить работу программы при x={π/16; π/8; 3π/16; π/4; 5π/16; 3π/8; 7π/16; π/2}.

Решение

Лист с исходными данными приведен на рисунке 1.6.

	A	B	C	D	E	F	G	H	I
1	Предельное значение слагаемого				0,0005				
2	X	0,19635	0,39270	0,58905	0,78540	0,98175	1,17810	1,37445	1,57080
3	Сумма ряда								
4	sin(x)								

Рисунок 1.6

Составим блок-схему алгоритма (рисунок 1.7). Здесь будет два цикла: внутренний и внешний. Во внутреннем организуем вычисление и вывод на лист очередной суммы, во внешнем - изменение значений x. Поскольку в условии не указано, сколько значений x следует использовать, создадим программу, которая будет вычислять суммы ряда для всех имеющихся на листе значений. Будем считать, что значения x должны быть введены в соседние ячейки, начиная с ячейки B2. То есть для вычислений будем считывать значения из всех ячеек, начиная с B2, пока не встретится пустая. Поскольку в слагаемых в знаменателе используются факториалы только нечетных чисел (1!, 3!, 5!...), чтобы факториал вычислялся правильно, добавлено вычисление пропускаемых значений Factor для четных чисел.

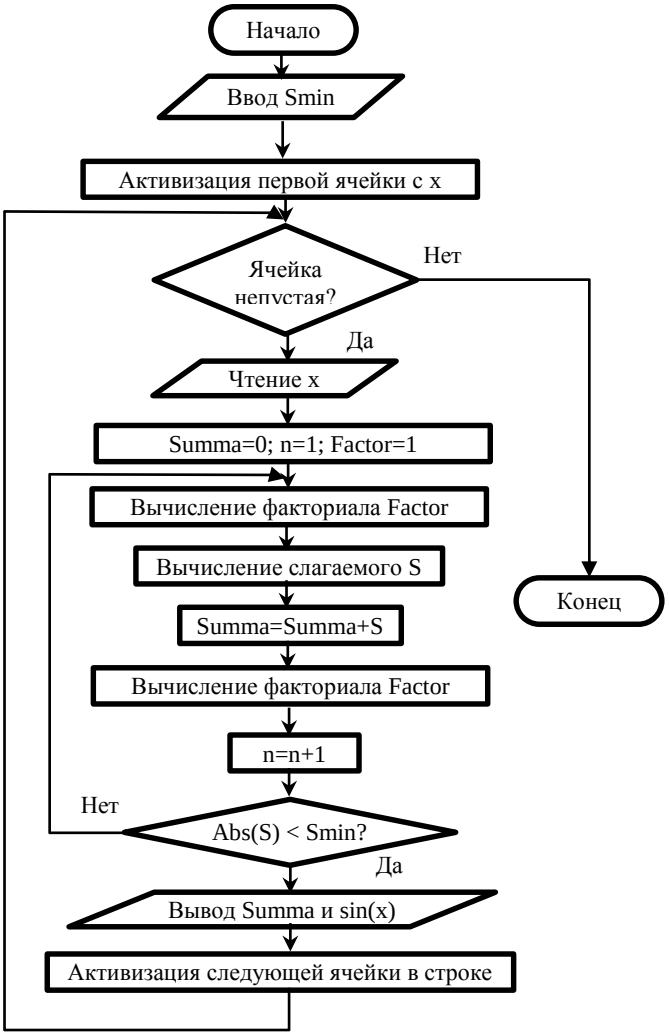


Рисунок 1.7

Пример программы приведен ниже.

```
Public Sub Primer1_2()
Dim n As Long, x As Double, S As Double, Smin As Double
Dim Factor As Long, Summa As Double
Dim I As Integer 'порядковый номер вычисляемой суммы
Smin = Range("E1") 'читаем из ячейки E1 значение минимального
 'слагаемого
Range("B2").Activate 'ячейка B2 становится текущей (активной)
I = 0
Do While ActiveCell.Value <> "" 'выполняем, пока активная ячейка
 'не окажется пустой
   I = I + 1
   x = ActiveCell.Value 'читаем из активной ячейки значение
   n = 1
   Factor = 1 'переменная для вычисления факториала в знаменателе
   Summa = 0
   Do
      Factor = Factor * (2 * n - 1) 'значение знаменателя в слагаемом
      S = ((-1) ^ (n - 1) * x ^ (2 * n - 1)) / Factor 'очередное слагаемое
      Summa = Summa + S
      Factor = Factor * 2 * n 'вычисление факториала для четных
'чисел
      n = n + 1
   Loop Until Abs(S) < Smin
   Cells(3, I + 1) = Round(Summa, 3) 'Вывод суммы ряда
   Cells(4, I + 1) = Round(Sin(x), 3) 'Вывод значения sin(x)
   ActiveCell.Offset(0, 1).Activate 'Сдвиг указателя активной ячейки
 'на 1 столбец вправо
Loop
End Sub
```

Для обращения к активной ячейке используется объект ActiveCell. Инструкция Range("B2").Activate вызывает для ячейки B2 метод Activate, тем самым устанавливая в нее указатель активной ячейки.

На рисунке 1.8 приведен лист с результатами работы программы.

	A	B	C	D	E	F	G	H	I
1	Предельное значение слагаемого				0,0005				
2	X	0,19635	0,39270	0,58905	0,78540	0,98175	1,17810	1,37445	1,57080
3	Сумма ряда	0,195	0,383	0,556	0,707	0,831	0,924	0,981	1
4	sin(x)	0,195	0,383	0,556	0,707	0,831	0,924	0,981	1

Рисунок 1.8

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Задача 1.1. Составить алгоритм в виде блок-схемы и программу для вычисления членов ряда, значение которых не более N_{max}. Члены ряда вычисляются в соответствии с указанным выражением. Результат вывести на лист MS Excel.

Вариант	Выражение	N_0	N_1	N_{max}
1	$N_{k+1}=2N_k+ N_{k-1}$	0	1	40000
2	$N_{k+1}=N_k+ 2N_{k-1}$	1	1	25000
3	$N_{k+1}=N_k*N_{k-1}$	1	2	36000
4	$N_{k+1}=N_k *2N_{k-1}$	1	1	42000
5	$N_{k+1}=2N_k+ 3N_{k-1}$	1	1	45000
6	$N_{k+1}=N_k+ 0,5N_{k-1}$	1	3	33000
7	$N_{k+1}=2N_k+ 2N_{k-1}$	1	1	30000
8	$N_{k+1}=3N_k- N_{k-1}$	0	2	25000
9	$N_{k+1}=2N_k- N_{k-1}$	1	3	15000
10	$N_{k+1}=3N_k- 0,5N_{k-1}$	1	1	10000

Задача 1.2. Составить алгоритм в виде блок-схемы и программу для вычисления суммы ряда при предварительно введенных нескольких значениях x. Процесс вычисления каждой суммы прекратить, когда очередное слагаемое по абсолютной величине окажется менее предельного. Сравнить полученные суммы со значением указанной функции. Исходные данные (значения x и предельное значение) получить из ячеек на листе MS Excel, результаты вычислений (значения сумм при соответствующих значениях x и функции), округленные до тысячных, вывести на лист. Проверить работу программы при любых 10 значениях x в заданном интервале.

Вариант	Сумма ряда	Предельн. значен.	Функция	x
1	$\sum_{n=1}^{\infty}\frac{(-1)^{n-1}x^n}{n}$	0,0025	$\ln(1+x)$	[0;3]
2	$\sum_{n=0}^{\infty}\frac{x^n}{n!}$	0,0035	e^x	[-1;1]
3	$\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n}}{(2n)!}$	0,0015	$\cos x$	[-π/4; π/4]
4	$\sum_{n=1}^{\infty}\frac{(-1)^n x^{2n+1}}{2n+1}$	0,001	$arctgx$	[-0,5;0,5]
5	$\sum_{n=0}^{\infty}\frac{(-1)^n (x+1)^n}{2n^{n+1}}$	0,0005	$\frac{1}{x+3}$	[-1;1]
6	$1+\sum_{n=1}^{\infty}\frac{(-1)^n 2^{2n-1}x^{2n}}{(2n)!}$	0,00035	$\cos^2 x$	[0; π/4]
7	$\sum_{n=0}^{\infty}\frac{2^n x^n}{n!}$	0,0002	e^{2x}	[-0,5;0,5]
8	$\sum_{n=0}^{\infty}\frac{x^{2n}}{(2n)!}$	0,0003	$\frac{e^x+e^{-x}}{2}$	[-1;1]
9	$\sum_{n=0}^{\infty}\frac{x^{2n+1}}{(2n+1)!}$	0,0004	$\frac{e^x-e^{-x}}{2}$	[-0,5;0,5]
10	$\sum_{n=0}^{\infty}\frac{x^{2n+1}}{2n+1}$	0,00025	$\frac{1}{2}\ln(\frac{1+x}{1-x})$	[-0,6;0,6]