МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ АРХИТЕКТУРНО-СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ

Кафедра прикладной математики

Основы программирования в VBA

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по курсу "ИНФОРМАТИКА"

для лабораторных и контрольных работ

для студентов

всех специальностей и направлений подготовки

Казань
2013

УДК 518.6
ББК 32.81
А95

А95 Основы программирования в VBA: методические указания по курсу "Информатика" для лабораторных и контрольных работ для студентов всех специальностей и направлений подготовки. / составители: Ф.Г. Ахмадиев, И.Г. Бекбулатов, Ф.Г. Габбасов. – Казань: Издательство Казанского государственного архитектурно-строительного университета, 2013. – 36 с.

Печатается по решению Редакционно-издательского совета Казанского государственного архитектурно-строительного университета

Методические указания предназначены для оказания помощи студентам при выполнении лабораторных и контрольных работ по курсу «Информатика» с применением новых информационных технологий в среде программирования Visual Basic for Applications. Рекомендовано для студентов всех специальностей и направлений подготовки.

Рецензент:
Доктор физико-математических наук,
заведующий кафедры высшей математики КГАСУ, профессор
Р.Б. Салимов

УДК 518.6
ББК 32.81

ВВЕДЕНИЕ

Целью данных методических указаний является выработка у студентов навыков алгоритмизации и программирования в VBA (Visual Basic for Applications) – языке программирования для приложения Excel. Методические указания содержат теоретический материал и примеры решения задач. Рассмотрены основные конструкции языка.

1. НАЧАЛЬНЫЕ СВЕДЕНИЯ

1.1. Назначение и применение ЭВМ

ЭВМ предназначена для автоматической обработки информации по заданным программам. Можно условно выделить три крупные области применения ЭВМ:

1. Информационные системы и средства коммуникации - поиск, обработка, хранение, передача информации, создание банка данных, расширение доступа к образованию, облегчению быта и т. д.

2. Автоматизация и управление различными видами работ человека - автоматизированные системы научных исследований (АСНИ), система автоматизации проектных работ (САПР), автоматизированные системы управления производством и технологическим процессом (АСУП и АСУТП), гибкие автоматизированные производства (ГАП) и др.

3. Математическое моделирование объектов и процессов разнообразной природы, вычислительный эксперимент.

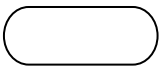
1.2. Этапы решения задач на ЭВМ

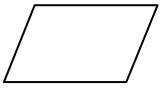
Основными этапами решения задач с применением ЭВМ являются:

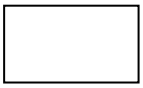
1. Постановка задачи и ее математическое описание.
2. Выбор численных методов решения (построение методики решения).
3. Алгоритмизация.
4. Программирование.
5. Отладка программы.
6. Непосредственное решение задачи на ЭВМ.
7. Обработка результатов расчета и их применение.

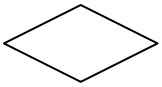
Разработка алгоритма - необходимый этап в процессе решения задачи на ЭВМ. **Алгоритм** – конечная последовательность точно определенных действий, приводящих к решению поставленной задачи.
Алгоритмы представляются в виде блок - схемы и в виде операторной записи при помощи символов - операторов или в виде перечисления этапов решения обычным текстом.

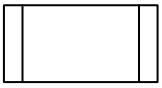
Блок - схема - графическое изображение алгоритма в виде последовательности блоков с помощью геометрических фигур. При составлении блок – схем используются следующие геометрические фигуры:


- начало и конец блок - схемы;


- блок ввода и вывода информации;


- блок вычисления арифметических выражений;


- блок проверки условий;


- блок вызова подпрограммы;


- ссылка (перенос блок - схемы на другую страницу).

Примеры алгоритмов смотрите на рисунках 2.1 - 2.7.

2. ВВЕДЕНИЕ В VBA
VBA, макросы, программирование в Excel.

Начиная с 1993 года, в состав Excel входит язык программирования VBA - Visual Basic для приложений (Visual Basic for Applications). Это язык программирования, основанный на Visual Basic, позволяет автоматизировать задачи Excel, является мощным дополнением к приложению, в более поздних версиях Excel пользователю доступна полнофункциональная интегрированная среда разработки. Можно создать VBA-код, повторяющий действия пользователя, и таким образом автоматизировать простые задачи. VBA позволяет создавать формы для общения с пользователем, более поздние версии позволяют использовать элементы объектно-ориентированного программирования.

2.1. Общая характеристика алгоритмического языка VBA

Ядром, основой среды программирования **VISUAL BASIC FOR APLICATIONS (VBA)** является язык программирования **VISUAL BASIC.** Большое множество различных версий языка Basic (такие как QB, QBasic, Quick Basic и т.д.) являются предшествующими языку **VISUAL BASIC. Я**зык **VISUAL BASIC** сохраняет все функциональные возможности практически всех предшествующих версий языка Basic.

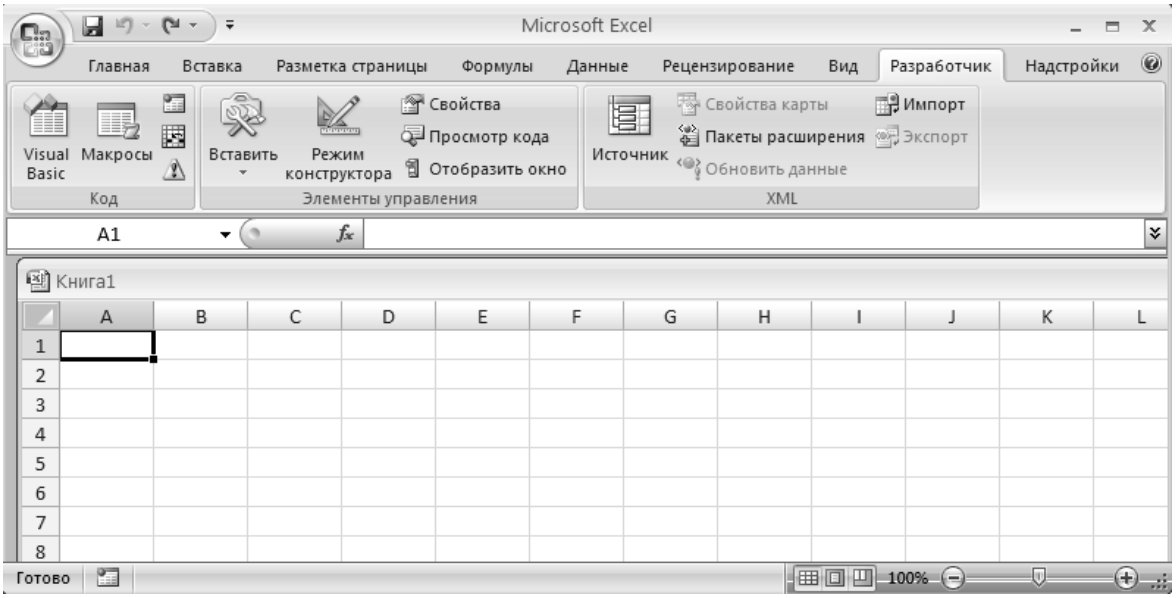
VBA (Visual Basic for Applications), является приложением языка программирования **Visual Basic** в Excel. Таким образом появилась возможность создавать программы находясь непосредственно в среде Excel. Программирование в Excel – это процесс создания макросов **Visual Basic,** встроенных в Excel.

Макросы (макрокоманды) – это программы, написанные на языке программирования.

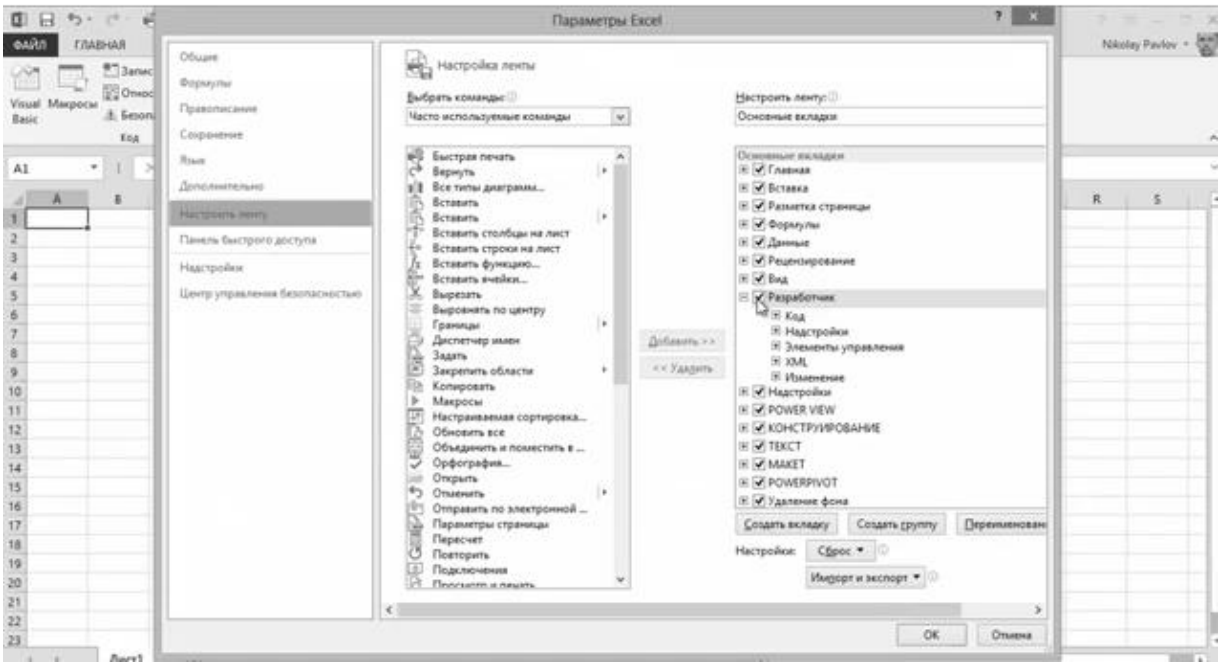
Разработка такой программы (Макроса) начинается с вставки модуля Visual Basic в активную книгу Excel . Для этого на ленте окна Excel нажимаем вкладку « **Разработчик** ». В результате появляется окно с редактором Visual Basic для создания модуля, и кнопка «**Макросы**», и колонка кнопок «**Запись макроса**», «**Относительные ссылки**», «**Безопасность макросов**». Для защиты Excel от макровирусов существует возможность выбора режима безопасности:

- полностью отключить макросы
- включить макросы при открытии документа
- доверять всем макросам, подписанным с использованием надёжных сертификатов.

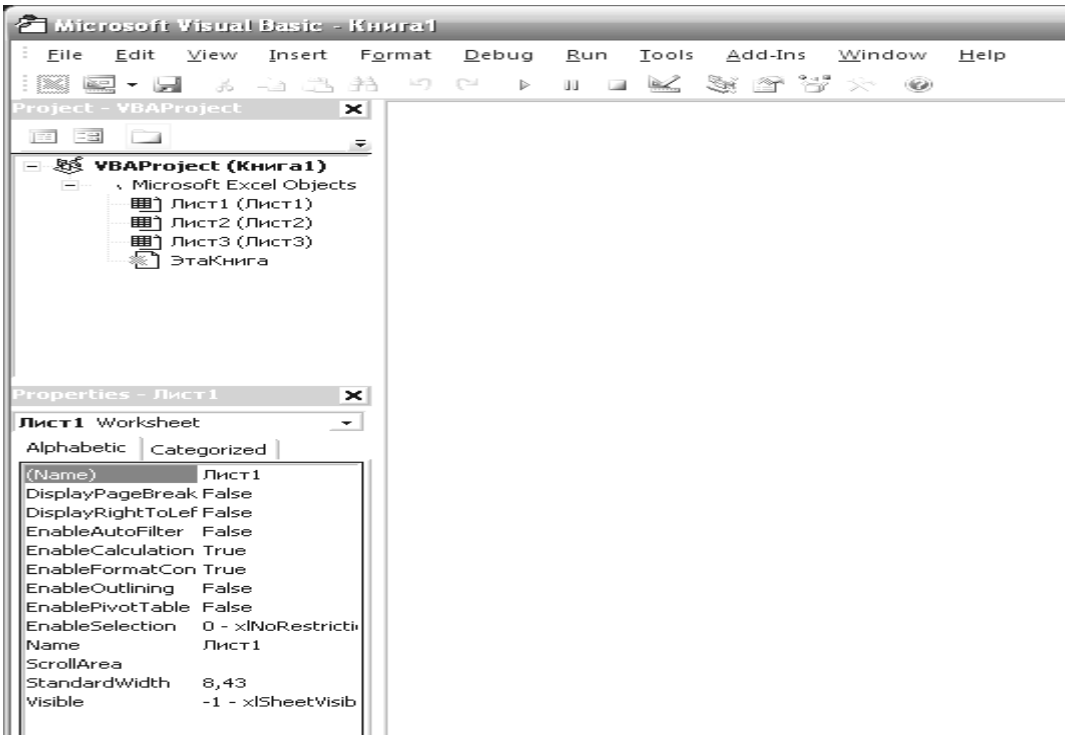
Кнопка «Макросы» предназначена для запуска, редактирования созданного модуля, сохранения его в нужной папке, удаления и т.д..



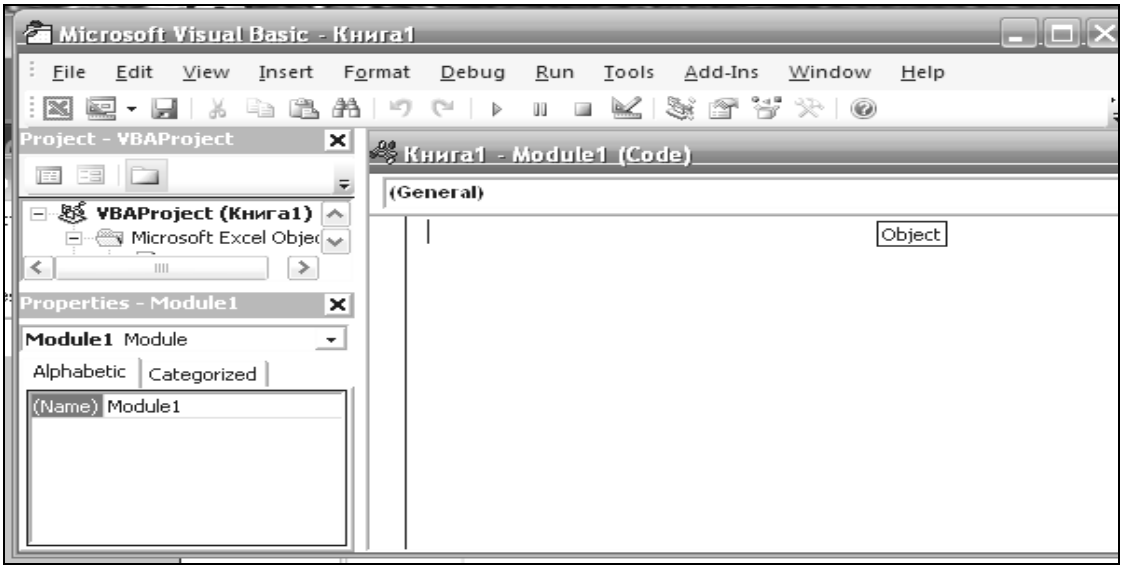
Если кнопка « Разработчик » отсутствует на ленте окна Excel,то в любом месте ленты щелкаем правой клавишей мыши, выбираем вкладку «Настройка ленты» и ставим галочку √ (левой клавишей мыши) напротив вкладки « Разработчик». После этого на ленте появится вкладка « Разработчик».



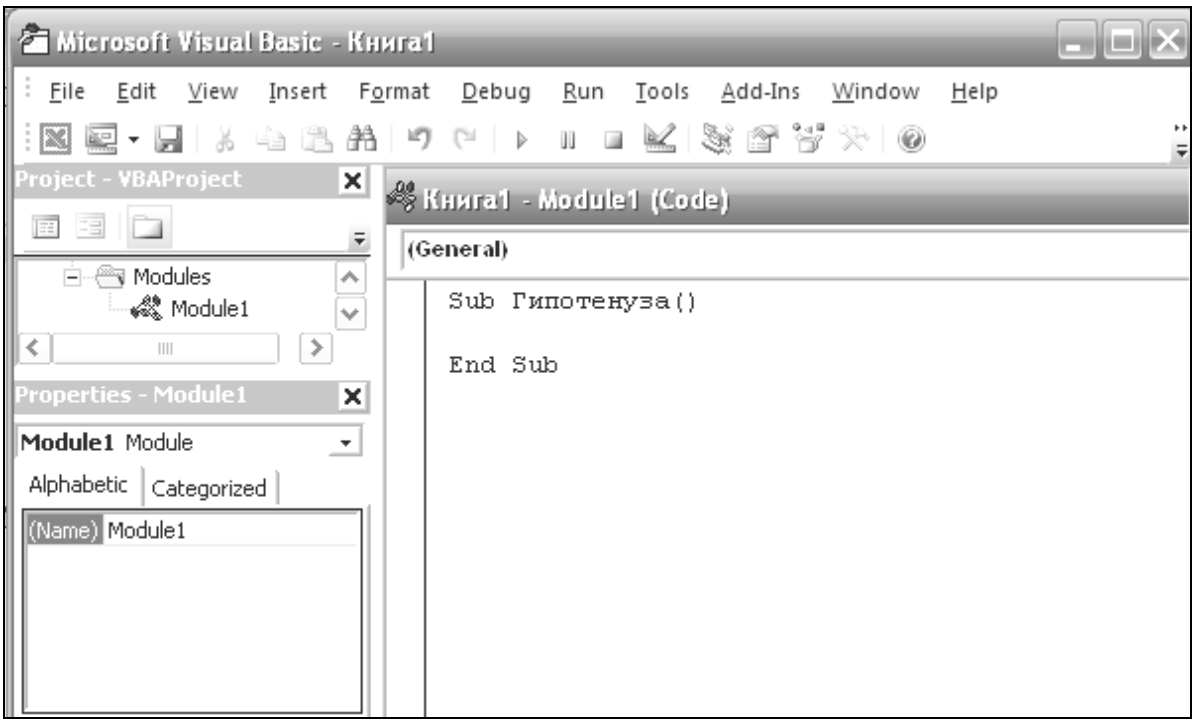
Для создания программы в Excel нажмем на вкладке «Разработчик» кнопку **Visual Basic**. Поверх окна Excel открывается окно редактора Visual Basic.



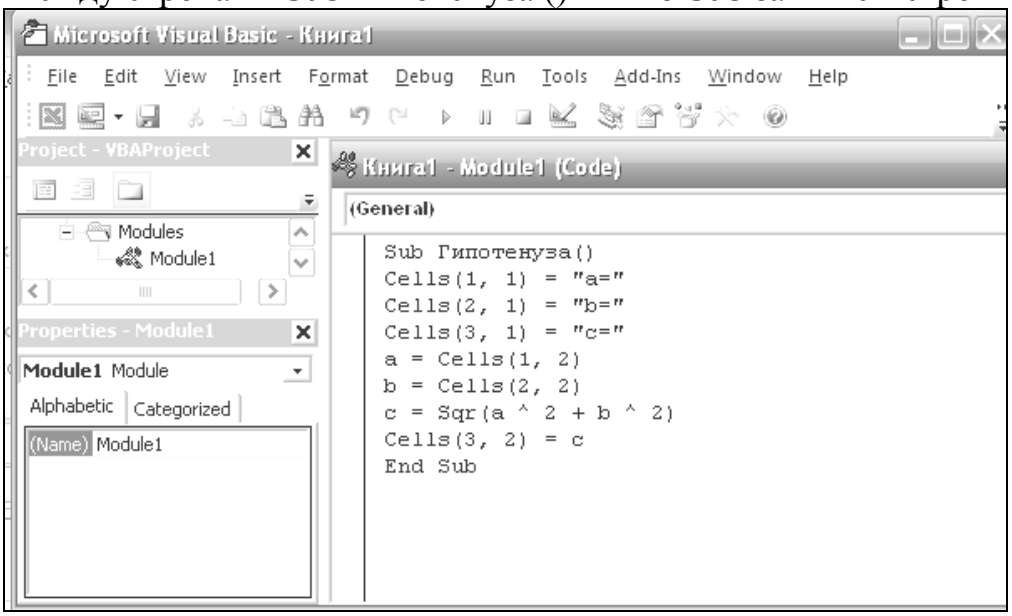
В этом окне можно будет выполнять различные действия: вводить и редактировать текст программы пользователя, производить отладку и выполнять программу. Программу, созданную пользователем, называют еще приложением, проектом или макросом. Она находится в книге Excel. Кликнем на строке **VBAProjekt (Книга1),** здесь **Книга1 –** название активной книги. Выполним **Insert > Module.** Откроется пустое окно – окно для вставки кода программы, соответствующее вставленному модулю с именем **Module1** .



Создадим нашу первую программу в Excel. Пусть, для примера программа должна вычислить длину гипотенузы по заданным двум катетам прямоугольного треугольника. Пусть значения длин катетов находятся в ячейках B1 и B2. Результат должен отобразиться в ячейке C2. Наберём в окне кода: Sub Гипотенуза и нажмем клавишу **Enter**.

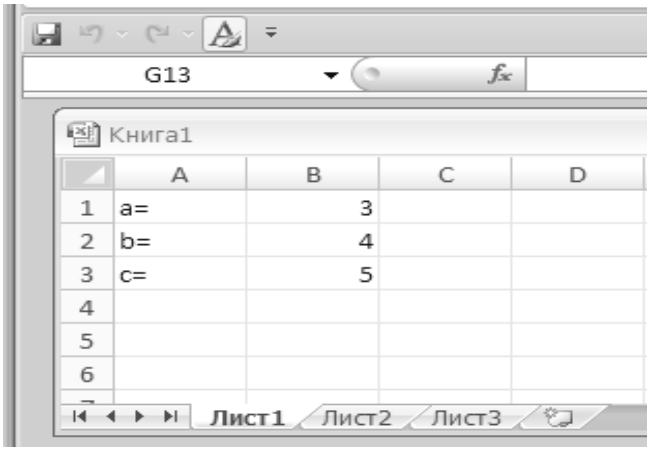


VBA автоматически представляет нам шаблон для создания Макроса. Между строками Sub Гипотенуза () и End Sub запишем строки.



Поясним смысл написанного кода программы.

Cells(1, 1) = "a=" : – в ячейку Excel A1 записывается текст "a=";
Cells(2, 1) = "b=": – в ячейку Excel A2 записывается текст "b=";
Cells(3, 1) = "c=": – в ячейку Excel A3 записывается текст "c=";
a = Cells(1, 2) – переменная *a* принимает значение, записанное в ячейке Excel B1;
b = Cells(2, 2) – переменная *b* принимает значение, записанное в ячейке Excel B2;
c = Sqr(a ^ 2 + b ^ 2) - – переменная *c* принимает значение, вычисленное по формуле: $c = \sqrt{a^2 + b^2}$;
Cells(3, 2) = *c* – в ячейку Excel B3 записывается значение переменной *c*.
На панели инструментов выполним: **Run > Run Sub/UserForm F5.**
Получим результат:



Другие команды:

“Selection.Copy” - означает “Выделение.Копировать”, то есть копируется выделенная область.

“Range(“C1”).Select” – “Диапазон(“C1”).Выделить” - переход к выделенной ячейке «C1».

«ActiveSheet.Paste» - «АктивныйЛист. Вставить» - вставляется в выделенную ячейку «C1» скопированное значение.

2.2. Символы языка VISUAL BASIC

1) двадцать шесть заглавных латинских букв A, B, C, D, ..., Z;
2) десять арабских цифр 0, 1, 2, ..., 8, 9;
3) знаки: “ . “ точка, “ , “ запятая, “ : “ двоеточие, “ ; “ точка с запятой, “ " “ кавычки, “ ‘ “ апостроф;

4) знаки арифметических операций: “ + “ сложение, “ – “ вычитание, “ * “ умножение, “ / “ деление, “ ^ “ возведение в степень;
5) круглые скобки: “ (“ открывающая скобка, “) “ закрывающая скобка;
6) “ ␣ ” пробел;
7) знаки операций отношения: “ < “ меньше, “ > “ больше, “ >= “ больше или равно, “ < > “ или “ > < “ не равно, “ <= ” меньше или равно, “ = “ равно;
8) знаки: “ $ “ знак доллара, “ & “ коммерческое “ и “, “ @ “ коммерческое “ эт “, “ \ “ косая черта слева направо, “ % “ процент, “ # “ номер, “ ? “ вопросительный знак, “ ! “ восклицательный знак;
9) буквы русского алфавита “ А, Б, В, ..., Я “.

2.3. Простейшие конструкции языка

Числа. Для отделения целой части числа от дробной используется точка (см. табл. 2.1).

Таблица 1

Примеры записи чисел на BASICe

Число	Запись на BASICe	Тип	Кол-во занимаемой памяти	Диапазон
3,46	3.46	Фиксированная запятая	4 байта	от 10^{-38} до 10^{38}
0,0058	58E-4	Плавающая запятая	4 байта	от 10^{-38} до 10^{38}
150000	15E 4	Плавающая запятая	4 байта	от 10^{-38} до 10^{38}
75	75 %	Целый	2 байта	от -32768 до 32767

Переменные. Для обозначения переменных используются имена, состоящие из буквы и цифр. Первым символом всегда является буква.

Например, A, A2, C. В качестве букв используются буквы латинского алфавита. Для целочисленных переменных, значениями которых являются целые числа, к имени добавляется знак % или &, для вещественных переменных обычной точности !, для вещественных переменных двойной точности #, для символьных переменных $.

Например, A%, A2%, C%.

Стандартные функции. Стандартные функции (см. таблицу 2.2) имеют аргумент, заключенный в круглые скобки. В качестве аргумента можно употреблять любое арифметическое выражение.

Таблица 2

Стандартные функции

№	Название функции	Математическое определение	Запись в VBA
1.	Синус	sin x	SIN(X)
2.	Косинус	cos x	COS(X)
3.	Тангенс	tg x	TAN(X)
4.	Арктангенс	arctg x	ATN(X)
5.	Показательная функция	e^x	EXP(X)
6.	Натуральный логарифм	ln x	LOG(X)
7.	Десятичный логарифм	lg x	LOG(X)/LOG(10)
8.	Знак сигнатуры	$\begin{cases} 1, \text{если } x > 0 \\ 0, \text{если } x = 0 \\ 1, \text{если } x < 0 \end{cases}$	SGN(X)
9.	Абсолютная величина	|x|	ABS(X)
10.	Квадратный корень	$\sqrt{x}$	SQR(X)
11.	Наибольшее целое, не превосходящее x	[x]	INT(X)
12.	Результат отбрасывания дробной части числа x	$\{x\}$	Fix(x)
13.	Датчик случайных чисел		RND(X)
14.	Число π	π	ATN(1)

Замечание. Аргумент функции RND можно опустить. Аргумент тригонометрической функции задается в радианах. Для перевода значения, заданного в градусах, в радианы можно использовать формулу:

<значение в радианах> = <значение в градусах> * π/180.

Для арктангенса значение угла находится в интервале (-π/2; π/2).

Для получения других обратных тригонометрических функций можно использовать формулы

arcsin(x)=arctg(x/$\sqrt{1-x^2}$),

arccos(x)=arctg($\sqrt{1-x^2}$)/x),

arcctg(x)=arctg(1/x).

2.4. Арифметические выражения

Для обозначения арифметических операций используются знаки:
“ + ” сложение, “ - “ вычитание, “ * ” умножение, “ / ” деление,
“ ^ ” возведение в степень, “ \ ” деление нацело, MOD остаток от деления целых чисел.

Если в арифметическом выражении имеется несколько различных арифметических операций, то порядок их выполнения задается правилами приоритета:

1. возведение в степень.
2. умножение, деление.
3. сложение, вычитание.

Примеры записи арифметических выражений

$\sin^2 x^3$	SIN(X^3)^2
$\sqrt[3]{x^4}$	X^(4/3)
$\frac{a^3 + e^{2*\cos(x)}}{\sqrt[3]{x^2 - y^3}}$	(a^3+EXP(2*COS(X)))/(X^2-Y^3)^(1/3)

2.5. Структура программы на языке VBA

Программа на **VBA** имеет модульную структуру. Основными модулями, имеющими наибольшее применение, являются SUB – главный модуль, FUNCTION – подпрограмма функция, SUB – подпрограмма – процедура. Любой модуль состоит из операторов – инструкций языка VBA. Эти инструкции (операторы) – представляют собой последовательности строк. В одной строке может содержаться один или несколько операторов, разделенных символом “ : ” двоеточие.

Программирование с использованием основных модулей.

Главный модуль имеет следующую структуру:
Sub ИМЯ()
Операторы
End Sub

Здесь ИМЯ - Идентификатор главного модуля.

Обособленную группу операторов, оформленную специальным образом, которую можно выполнять многократно, обращаясь к ней из различных мест главного модуля, называют подпрограммой. Чтобы подпрограмма при обращении к ней выполнялась каждый раз с новыми данными, ее нужно составить в общем виде, а исходные данные для работы передавать в переменные подпрограммы перед обращением к ней.

Модуль FUNCTION - подпрограмма-функция.

Это подпрограмма имеет следующую структуру:

```
FUNCTION имя [(список параметров) STATIC]
[Операторы подпрограммы]
имя = выражение
END FUNCTION
```

Для обращения к функции используют указатель функции содержащую имя функции и в скобках конкретные значения ее аргументов.

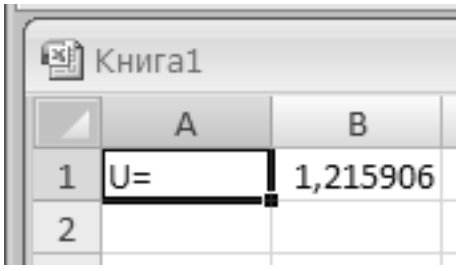
Результат выполнения подпрограммы- функции в главную программу передается через локальную переменную, имя которой должно совпадать с именем подпрограммы.

Пример 1. Вычислить $u = r(\frac{1}{2}) + r(\frac{1}{3}) + r(\frac{1}{4})$, где $r(x) = x + \frac{x^2}{2^2} + \frac{x^3}{3^2} + ... = \sum_{k=1}^{\infty} \frac{x^k}{k^2}$.

Сумму вычислить с точностью 0,0001.Для этого необходимо суммировать до тех пор, пока очередной член суммы не станет меньше заданной точности.

Используем подпрограмму - функцию.

```
Function R(X)
S = 0: K = 1
While Abs(X ^ K / K ^ 2) > 0.0001
S = S + X ^ K / K ^ 2
K = K + 1
Wend
R = S
End Function
Sub Pr1()
U = R(1 / 2) + R(1 / 3) + R(1 / 4)
Cells(1, 1) = "U="
Cells(1, 2) = U
End Sub
```



Пример 2.Вычислить скалярное произведение векторов:
$(a_1, a_2, ..., a_k)$ и $(b_1, b_2, ..., b_k)$;
$(c_1, c_2, ..., c_m)$ и $(d_1, d_2, ..., d_m)$.

Используем подпрограмму, вычисляющую скалярное произведение векторов $(x_1, x_2, ..., x_n)$ и $(y_1, y_2, ..., y_n)$:

$$s = \sum_{i=1}^{n} x_i y_i .$$

```
Sub Pr2()
Rem "Основная программа"
k = Cells(2, 1): m = Cells(2, 1)
ReDim a(1 To k), b(1 To k), C(1 To m), D(1 To m)
For i = 1 To k
a(i) = Cells(3 + i, 1): b(i) = Cells(3 + i, 2)
Next
For i = 1 To m
C(i) = Cells(3 + i, 3): D(i) = Cells(3 + i, 4)
Next
S1 = SP(a(), b(), k)
S2 = SP(C(), D(), m)
Cells(2, 5) = "S1"
Cells(2, 6) = "S2"
Cells(3, 5) = S1
Cells(3, 6) = S2
End Sub
Function SP(X(), Y(), N)
Rem "Подпрограмма"
S = 0
For i = 1 To N
S = S + X(i) * Y(i)
Next
SP = S
End Function
```

Pr11.xlsm [Последнее сохранение пользователем]

	A	B	C	D	E	F	G
1	k	m					
2	4	5			S1	S2	
3	a	b	c	d	20	120	
4	1	4	2	10			
5	2	3	4	8			
6	3	2	6	6			
7	4	1	8	4			
8			10	2			
9							

Модуль SUB - подпрограмма общего вида (процедура)

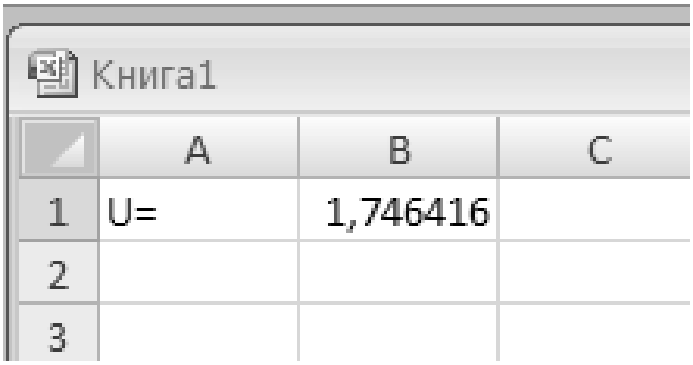
Это подпрограмма имеет следующую структуру:
SUB имя [(список параметров)] [STATIC]
Операторы подпрограммы
END SUB

Для обращения к подпрограмме общего вида служит оператор
[CALL] имя подпрограммы [(список аргументов)].

Результаты выполнения подпрограммы - процедуры передаются в главную программу через локальные переменные, указанные в списке параметров. Поэтому, в отличие от подпрограммы – функции, из подпрограммы – процедуры в главную программу можно передать большее количество данных.

Используем подпрограмму общего вида для решения примера 1.

```
Sub Pr1_main()
Rem "Основная программа"
Call P(1 / 2, R1)
Call P(1 / 2, R2)
Call P(1 / 2, R3)
U = R1 + R2 + R3
Cells(1, 1) = "U="
Cells(1, 2) = U
End Sub
```




```
Sub P(X, R)
Rem "Подпрограмма процедура"
R = 0: K = 1
While Abs(X ^ K / K ^ 2) > 0.0001
R = R + X ^ K / K ^ 2
K = K + 1
Wend
```

Для решения примера 2.

```
Sub Pr2_procedura()
Rem "Основная программа"
k = Cells(2, 1): m = Cells(2, 1)
ReDim a(1 To k), b(1 To k), C(1 To m), D(1 To m)
For i = 1 To k
a(i) = Cells(3 + i, 1): b(i) = Cells(3 + i, 2)
Next
For i = 1 To m
C(i) = Cells(3 + i, 3): D(i) = Cells(3 + i, 4)
Next
```

```
Call SP(a(), b(), k, S1)
Call SP(C(), D(), m, S2)
Cells(2, 5) = "S1"
Cells(2, 6) = "S2"
Cells(3, 5) = S1
Cells(3, 6) = S2
End Sub

Sub SP(X(), Y(), N, S)
Rem "Подпрограмма"
S = 0
For i = 1 To N
S = S + X(i) * Y(i)
Next
End Sub
```

Pr11.xlsm [Последнее сохранение пользователем]

	A	B	C	D	E	F	G
1	k	m					
2	4	5			S1	S2	
3	a	b	c	d	20	120	
4	1	4	2	10			
5	2	3	4	8			
6	3	2	6	6			
7	4	1	8	4			
8			10	2			
9							

2.5.1. Программы линейной структуры

Линейная программа должна состоять из следующих операторов (см. рис. 2.1): ввода данных, присваивания, вывода (печать) результатов расчета. Линейный вычислительный процесс сводится к последовательным вычислениям арифметических выражений, причем последовательность вычислений полностью соответствует порядку записи математических зависимостей в постановке задачи.

Оператор-комментарий. Для облегчения восприятия и большей наглядности программы в нее целесообразно включать комментарии, которые поясняют работу отдельных частей программы, характеризуют используемые переменные и т.д.

Для записи комментариев используется **оператор REM** (REMARK). В этом операторе за словом REM могут быть записаны любые символы **VISUAL BASIC**. Включение операторов REM в программу никак не влияет на ее выполнение.

Операторы ввода и вывода.

Оператор ввода служит для задания исходных данных при выполнении программы. Операторы вывода позволяют получить результаты выполнения программы.

Ввод и вывод может осуществляться в нескольких формах: обмен данными между программой и ячейками Excel и между программой и стандартным окном Windows.

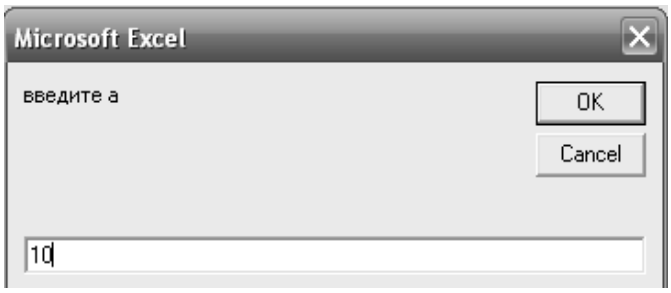
Ввод и вывод с ячейками Excel показаны на следующих примерах (подробно описано в п. 2.1, стр. 8.9):

a = Cells(1, 2) – ввод значения переменной *a* из ячейки B1;

Cells(1, 2)= a – вывод значения переменной *a* в ячейку B1;

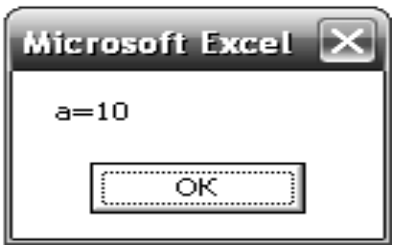
Ввод и вывод при помощи стандартного окна Windows показаны на следующих примерах:

a =InputBox("введите a") – ввод значения для переменной a.

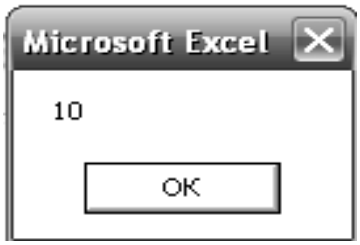


Сообщение "введите a" появляется в окне для подсказки, чтобы человек знал, для какой переменной требуется ввести его значение. Здесь в качестве значения для переменной “a” введено число 10.

MsgBox ("a="&a) – вывод сообщения "a=" и значения переменной a,



или: **MsgBox a**.



Оператор присваивания. Оператор присваивания служит для присваивания переменной значения арифметического выражения и имеет вид **LET V = <арифметическое выражение>**

(LET можно опустить), где V - переменная, которой присваивается значение.

При выполнении оператора присваивания вычисляется выражение в правой части и присваивается переменной в левой части V.

Оператор END. Оператор END указывает на физический конец программы. При выполнении оператора END закрываются все открытые файлы и останавливается выполнение программы.

Пример 3. Вычислить значение функции

$$y = \frac{\cos(x^2 - \sqrt{t}) + \sin(x - \sqrt[3]{t})}{\sin^2(t - \sqrt[4]{x}) + 1{,}731},$$

где $x = e^{-\sqrt{n/m}}$, $t = \ln\frac{m}{\sqrt{n}} + 7{,}21$, m - номер варианта, n - номер группы.

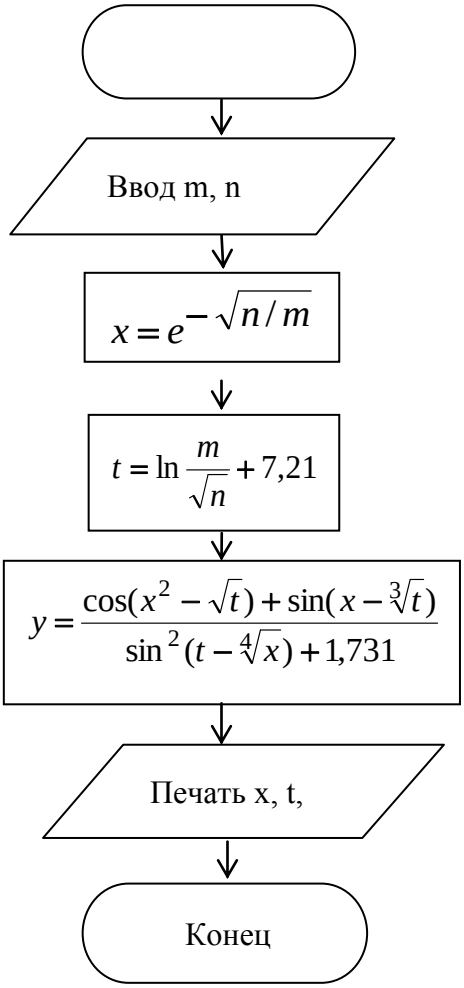
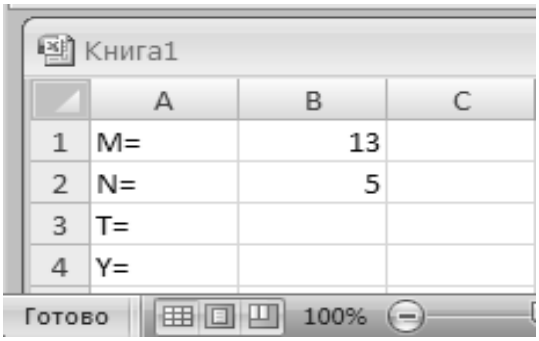


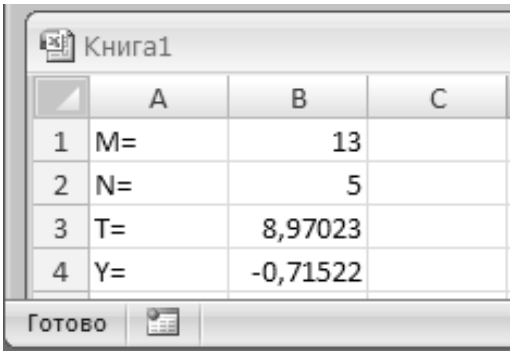
Рис. 2.1. Блок - схема линейного вычислительного процесса

Программа для примера 3 (см. рис. 2.1.):

```
Sub Pr3()
M=Cells(1,2)
N=Cells(2,2)
X=EXP(-(N/M)^(1/2))
T=LOG(M/SQR(N))+7.21
Y=(COS(X^2-SQR(T))+SIN(X-T^(1/3)))/(SIN(T-X^(1/4))^2)+1.731)
Cells(3,2)=T
Cells(4,2)=Y
END
```

Предварительно в Excel необходимо создать лист следующего вида:





После ввода текста программы макроса ее запуска результат появится в ячейках B3 и B4.

Определение функций пользователя.

Общая структура построения функций пользователя

Function **F(аргумент $_1$, аргумент $_2$,…, аргумент $_n$)**
F = <арифметическое выражение>
End Function
где **F**- имя функции; **<аргументы>** - простые переменные - формальные аргументы (формальные параметры) функции; **<арифметическое выражение>** - формула, по которой вычисляется функция.

Арифметическое выражение в правой части должно содержать хотя бы один из формальных параметров, но может содержать также и другие переменные, общие для всей программы.

Пример 4. Вычислить значение функции

$$z = \frac{f(m/n, -2n) + f(3(m+n), -5,5m)}{2(m+n)}, \qquad \text{где } f = \cos(x^2 + y^2).$$

Здесь m - номер варианта, n –номер группы.

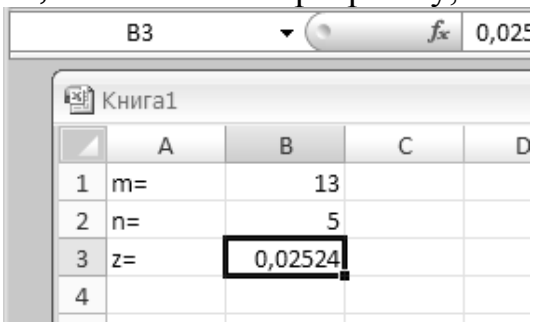
1- й вариант программы:

В окно для вставки кода записываем:

```
Function F(X, Y)
F = Cos(X ^ 2 + Y ^ 2)
End Function
```

```
Sub main()
m = Cells(1, 2)
n = Cells(2, 2)
Z = (F(m / n, -2 * n) + F(3 * (m + n), -5.5 * m)) / (2 * (m + n))
Cells(3, 1) = "z="
Cells(3, 2) = Z
End Sub
```

В ячейки Excel A1 и B1 записываем m=, 13 соответственно, и в A2 и B2 - n=, 5. Выполним программу, нажав F5. Получим:

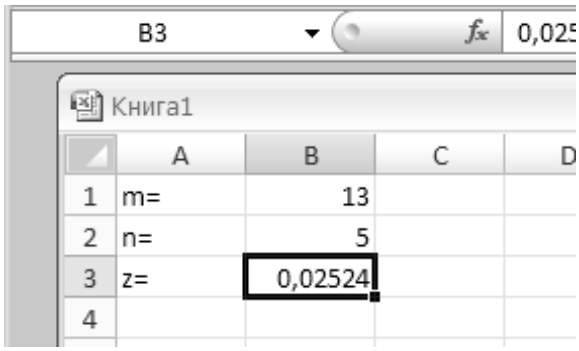


2- й вариант программы:

В окно для вставки кода записываем:

```
Function F(X, Y)
F = Cos(X ^ 2 + Y ^ 2)
End Function
```

В ячейки Excel A1 и B1 записываем m=, 13 соответственно, и в A2 и B2 - n=, 5. Далее, в ячейку Excel B3 записываем формулу:
=(F(B1/B2;-2*B2)+F(3*(B1+B2);-5,5*B2))/2/(B1+B2)
При записи этой формулы можно пользоваться технологией вызова функций Excel (нажав кнопку ***fx*** и выбрав категорию – функции пользователя). В результате в ячейке B3 появится результат: 0,02524.



2.5.2. Программы ветвящейся структуры на языке VBA

Для составления разветвляющихся программ используются операторы безусловной и условной передачи управления (см. рис. 2.2). В разветвляющихся вычислительных процессах последовательность выполнения операций заранее не определена и ставится в зависимость от результатов проверки заданных условий.

Оператор безусловного перехода.

Общий вид оператора

GO TO m, где m метка. Этот оператор передает управление первому оператору в строке с меткой m.

Условные операторы. Условные операторы служат для изменения порядка выполнения операторов в зависимости от выполнения или невыполнения какого - либо условия.

Условные операторы могут использоваться для организации циклов и ветвлений.

Общий вид условных операторов

1) IF «условие» THEN (или GO TO) m;
2) IF «условие» THEN «действие»;
3) IF «условие» THEN «действие 1» ELSE «действие 2»;
4) IF «условие 1» THEN
«Блок 1»
ELSEIF «условие 2» THEN
«Блок 2»
. . .
ELSE
«Блок n»
END IF

где условие имеет вид:

(арифметическое выражение 1) θ (арифметическое выражение 2)

θ - одна из операций отношения <, <=, >, >=, =, <>;

m - метка; действие - любой оператор **BASIC**а, в том числе это может быть другой условный оператор.

Сочетание THEN IF позволяет "вкладывать" условия друг в друга. Например,

IF Q >= 3.5 THEN IF Q < 7.4 THEN 20

Действие условного оператора заключается в следующем:

если условие удовлетворяется, то в первом случае осуществляется переход к строке с меткой m=20, во втором и в третьем случаях выполняется оператор, следующий за THEN.

Если условие не удовлетворяется, то в первом случае осуществляется переход к оператору, следующему за условным. Во втором случае осуществляется переход к первому оператору следующей строки, т. е. все операторы в строке, следующие за условным оператором, при этом игнорируются. В третьем случае выполняется действие, записанное после ключевого слова ELSE.

Возможна одновременная проверка нескольких условий, соединенных словами:

OR - условие считается выполненным, если выполняется хотя бы одно из проверяемых условий;

AND - условие выполнено, если выполняются все проверяемые условия одновременно;

XOR - условие выполнено, если выполняется ровно одно из двух проверяемых условий.

Пример 5. Вычислить $y = \begin{cases} x^2 + 3x - 7, & \text{если } x < 0, \\ 2x - 1, & \text{если } 0 <= x < 1, \\ e^x, & \text{если } x >= 1 \end{cases}$

где $x = 3\cos\frac{m}{n}$.

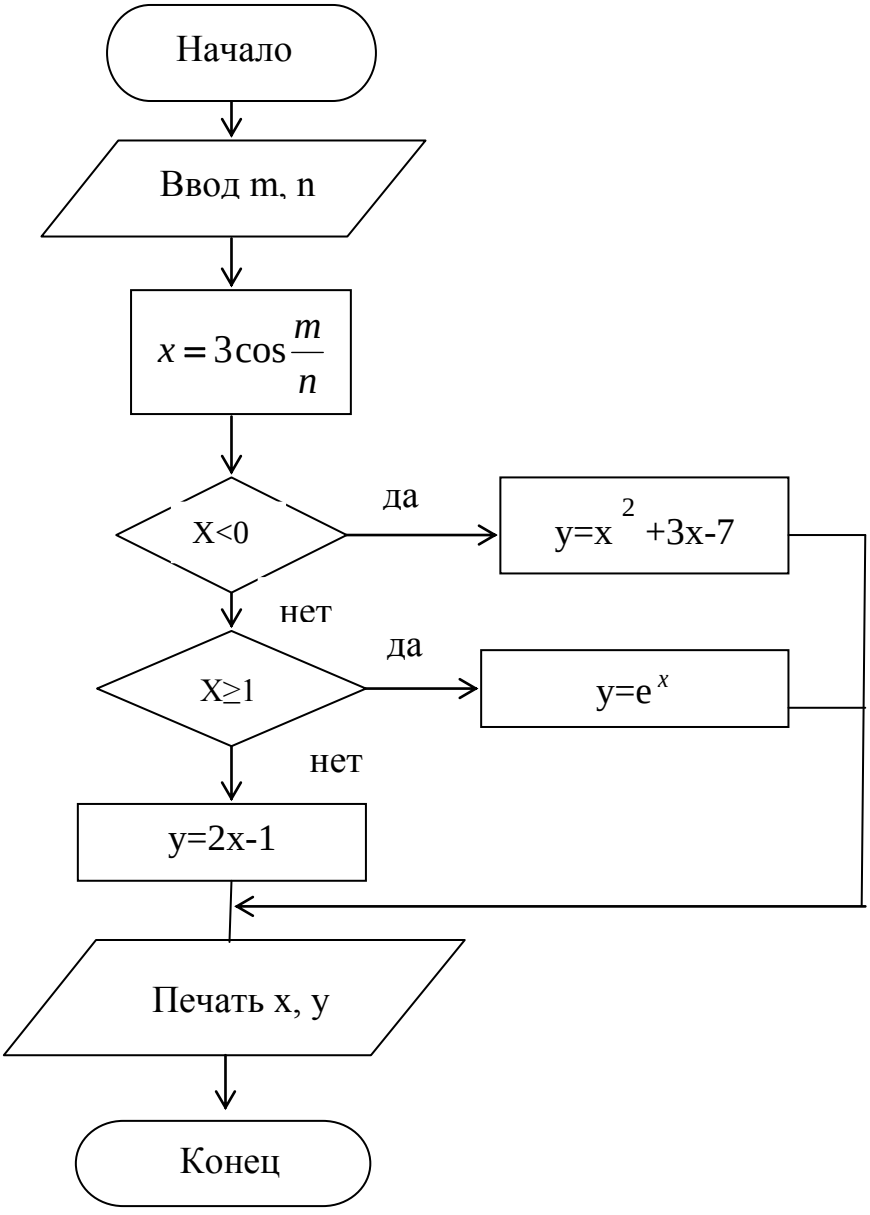


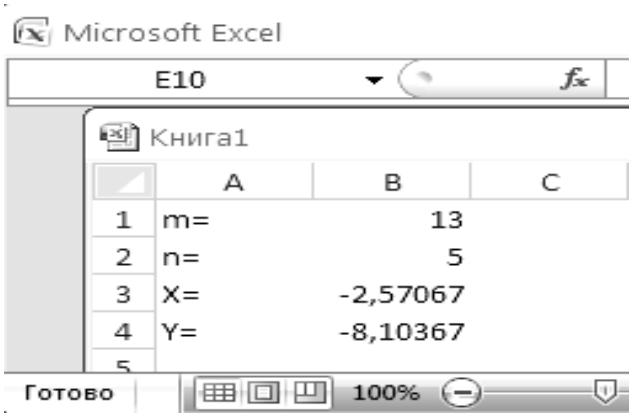
Рис. 2.2. Блок - схема разветвляющегося вычислительного процесса

Программа для примера 5 (см. рис. 2.2.):
Предварительно в Excel необходимо создать лист входных данных:

Код программы:

```
Sub Pr5()
m = Cells(1, 2)
n = Cells(2, 2)
X = 3 * Cos(m / n)
If  X < 0 Then
Y = X ^ 2 + 3 * X - 7
ElseIf  X >= 1 Then
Y = Exp(X)
Else
Y = 2 * X - 1
End If
Cells(3, 1) = "X=": Cells(3, 2) = X
Cells(4, 1) = "Y=": Cells(4, 2) = Y
End Sub
```

После ввода текста программы макроса и ее запуска результат появится в ячейках диапазона A3:B4.



2.5.3. Программы циклической структуры на языке VISUAL BASIC.

Программы циклической структуры это такие программы, в которых какая-то группа операторов многократно повторяется. Эта группа операторов, оформленная специальным образом, называется циклом. Количество повторений определяется либо параметрами цикла, либо условием, заданным вне цикла. Многократное повторение выполняется за счет передачи управления на начало этой группы операторов. Циклические

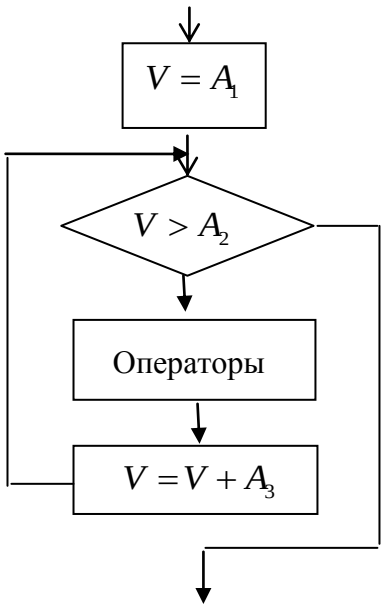
алгоритмы применяются при решении задач на табулирование функций (составление таблицы значений функции), на вычисление суммы и произведений, при обработке массивов.

Операторы цикла. Для организации циклов в VISUAL BASIC применяются операторы FOR и NEXT. Общий вид операторов

FOR V = A_1 **TO** A_2 **STEP** A_3
Операторы
NEXT V

где v - любая неиндексированная переменная - управляющая переменная цикла; A_1, A_2, A_3 - начальное и конечное значения и шаг изменения управляющей переменной цикла - любые арифметические выражения. Если A_3=1, то конструкцию STEP A_3 можно опустить. Операторы, расположенные между операторами FOR и NEXT, образуют тело цикла и выполняются многократно.

Блок- схема оператора FOR … NEXT



Выполнение цикла, образованного операторами FOR и NEXT, показано в этой блок-схеме.

Есть и другие операторы цикла.
Например:
WHILE условие
Операторы
WEND , здесь операторы выполняются, пока выполняется условие

Пример 6. Составить таблицу значений функции *y* при изменении значений переменной *x* на отрезке [-2,2] с шагом h=0,5:

$$y = \frac{e^{-x} + 5m}{x + n}$$

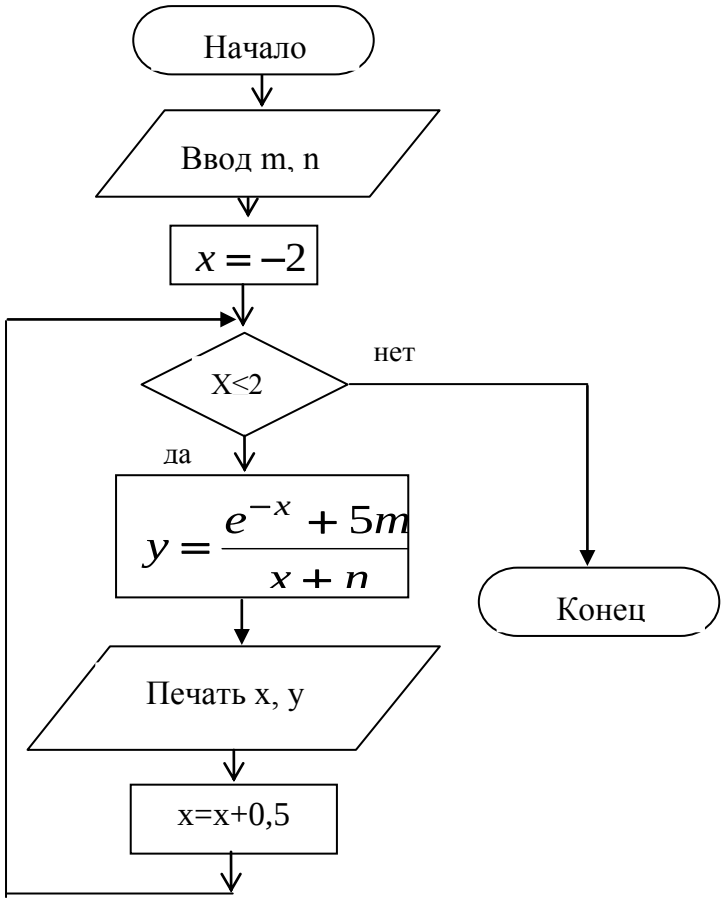
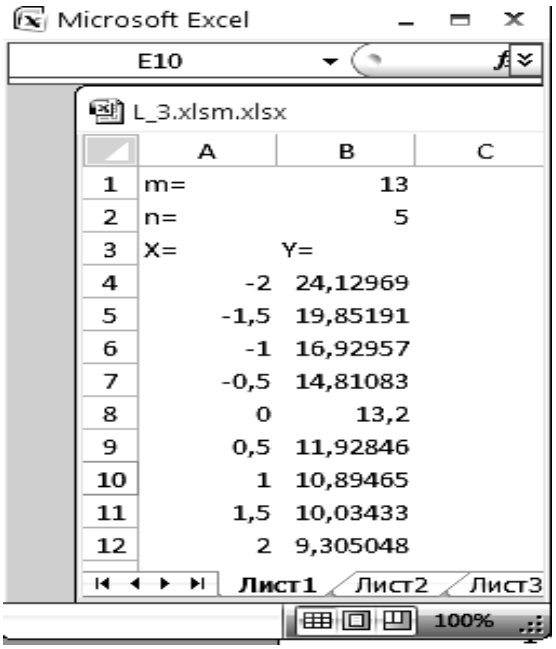


Рис. 2.3. Блок - схема табулирования функции *y=f(x)*

Программа для примера 6:

```
Sub Pr6()
m = Cells(1, 2):n = Cells(2, 2)
Cells(3, 1) = "X=": Cells(3, 2) = "Y="
i = 0
For X = -2 To 2 Step 0.5
Y = (Exp(-X) + 5 * m) / (X + n)
Cells(4 + i, 1) = X:Cells(4 + i, 2) = Y
i = i + 1
Next X
End Sub
```



<u>Пример 7</u>. Составить таблицу значений функции **z=f(x;y)** при изменении значений переменной **x** на отрезке **[*a*,*b*]** с шагом ***h*** , переменной ***y*** на отрезке **[*c*,*d*]** с шагом ***l***: $z = \frac{x^2 - y^2 + m}{(x^2 + y^2) * n}$, где ***a***=1, ***b***=2, ***h***=0,2; ***c***=2, ***d***=4, ***l***=0,2.

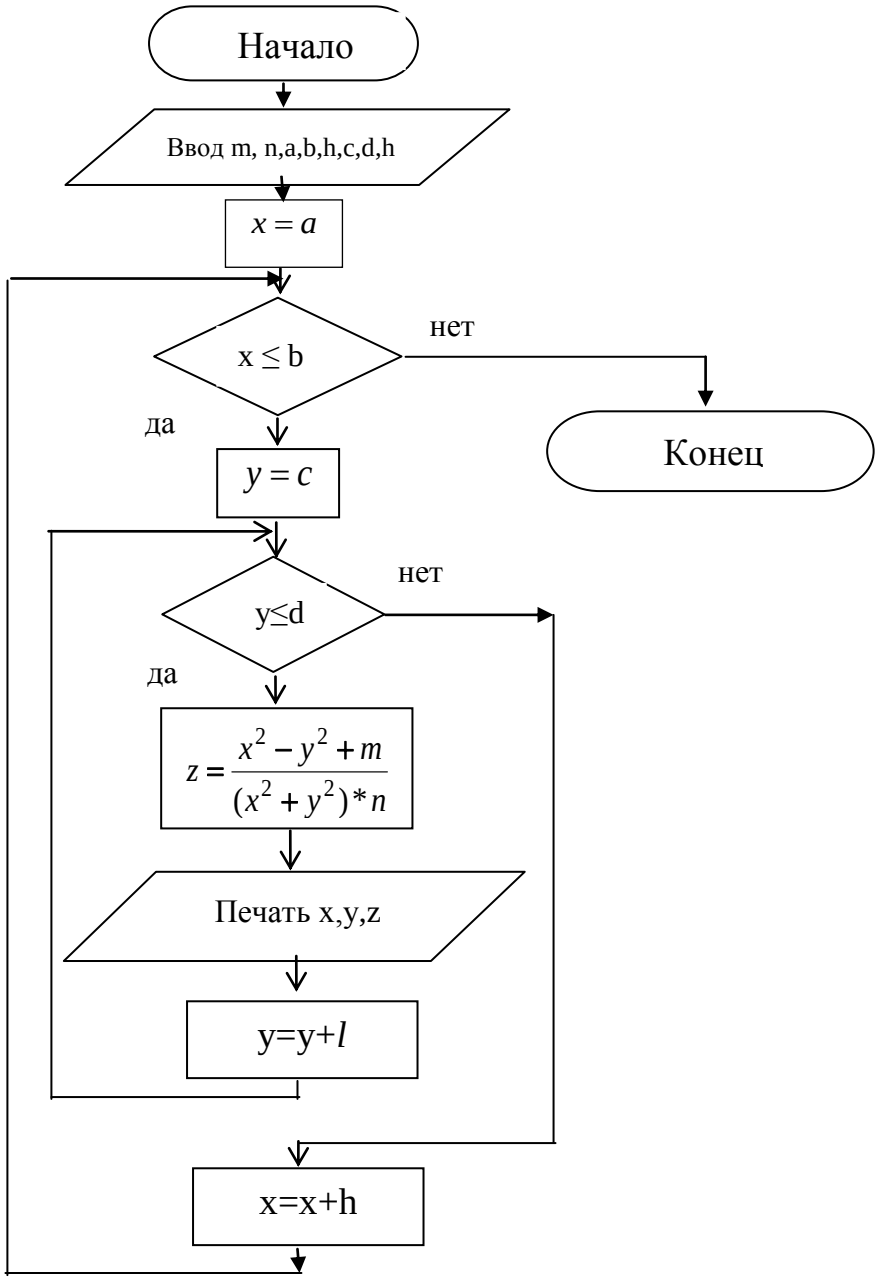


Рис. 2.4. Блок - схема программы табулирования функции *z=f(x,y)*

Программа для примера 7:

```
Sub Pr7()
m = Cells(1, 2): n = Cells(2, 2)
a = Cells(4, 1): b = Cells(4, 2): h = Cells(4, 3)
c = Cells(6, 1): d = Cells(6, 2): l = Cells(6, 3)
Cells(7, 1) = "  X/Y  "
j = 2
For y = c To d + 0.01 * l Step l
Cells(7, j) = y: j = j + 1
Next y
i = 0
For x = a To b + 0.01 * h Step h
Cells(8 + i, 1) = x
j = 2
For y = c To d + 0.01 * l Step l
Z = (x ^ 2 - y ^ 2 + m) / (x ^ 2 + y ^ 2) / n
Cells(8 + i, j) = Z: j = j + 1
Next y: i = i + 1
Next x
End Sub
```

Книга1

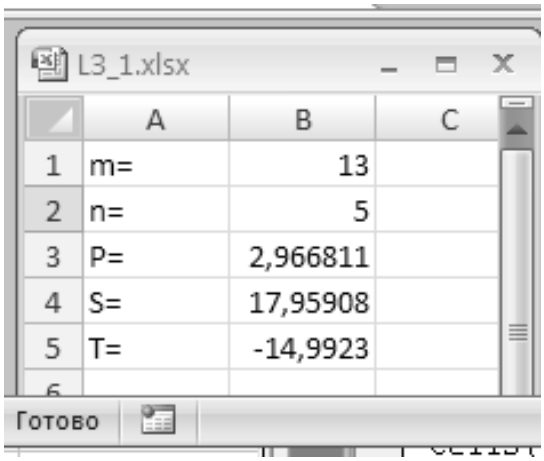
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	m=	13											
2	n=	5											
3	a	b	h										
4	1	2	0,2										
5	c	d	l										
6	2	4	0,2										
7	X/Y	2	2,2	2,4	2,6	2,8	3	3,2	3,4	3,6	3,8	4	
8	1	0,4	0,313699	0,243787	0,186598	0,139367	0,1	0,066904	0,038854	0,0149	-0,0057	-0,02353	
9	1,2	0,383824	0,305732	0,241111	0,187317	0,142241	0,104215	0,071918	0,044308	0,020556	-1,6E-16	-0,01789	
10	1,4	0,367785	0,297647	0,238342	0,188073	0,145306	0,108759	0,077377	0,050296	0,02681	0,006341	-0,01158	
11	1,6	0,352439	0,28973	0,235577	0,188841	0,148462	0,113495	0,083125	0,056657	0,033505	0,013176	-0,00474	
12	1,8	0,338122	0,282178	0,232889	0,1896	0,151625	0,118301	0,089021	0,063243	0,040494	0,020362	0,002495	
13	2	0,325	0,275113	0,230328	0,190335	0,15473	0,123077	0,094944	0,069923	0,047642	0,027766	0,01	

Готово

Пример 8. Вычислить $t = P - S$ где $P = \prod_{k=1}^{15} \frac{n^2}{\sqrt{mk^2+1}}$, $S = \sum_{i=2}^{20} \frac{n^2}{\sqrt{mk^2+1}}$

Программа для примера 8 (см. рис. 2.5.):

```
Sub Pr8()
m = Cells(1, 2): n = Cells(2, 2)
Rem "Вычисление произведения"
P = 1
For k = 1 To 15
P = P * n ^ 2 / Sqr(m * k ^ 2 + 1)
Next k
Rem "Вычисление суммы"
S = 0
For k = 2 To 20
S = S + n ^ 2 / Sqr(m * k ^ 2 + 1)
Next k
T = P - S
Cells(3, 1) = "P=": Cells(3, 2) = P
Cells(4, 1) = "S=": Cells(4, 2) = S
Cells(5, 1) = "T=": Cells(5, 2) = T
End Sub
```



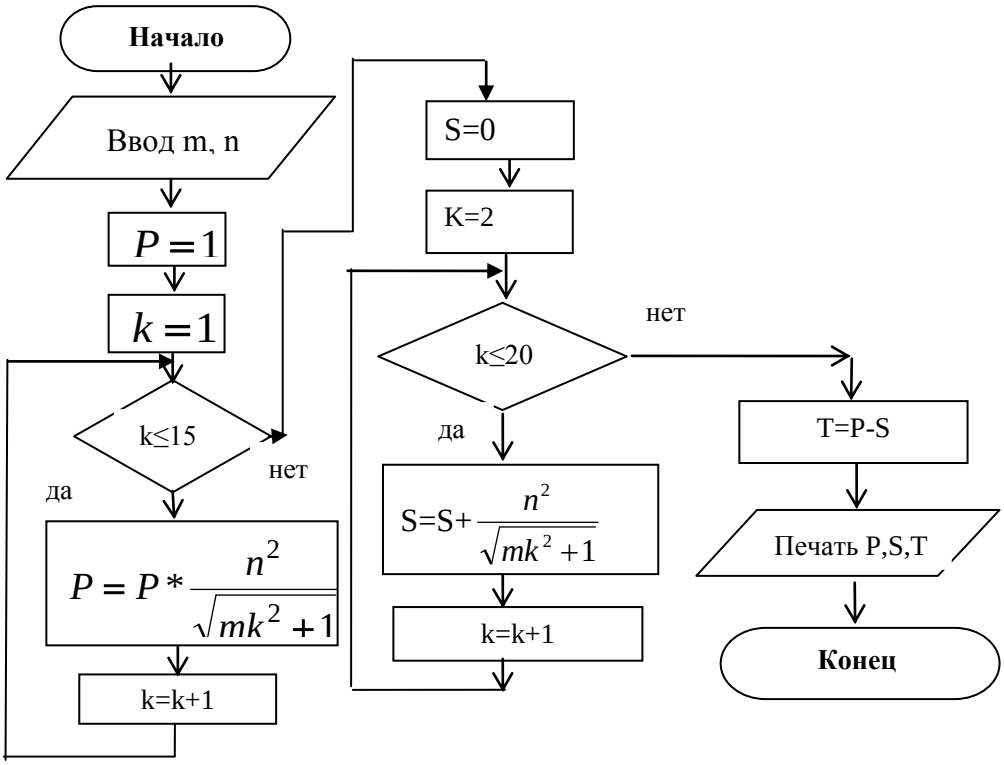


Рис. 2.5. Блок - схема циклического вычислительного процесса

2.5.4. Программирование с использованием массивов.

Массивом называется последовательность элементов, обозначаемых одним именем, где каждый элемент имеет свое определенное место. Это место в последовательности определяется комбинацией индексов. Количество индексов определяет **размерность** массива. Произведения диапазонов изменения индексов определяет **размер** массива - количество элементов в массиве. Например, вектор $a_1, a_2, ..., a_n$ – одномерный массив, так как для определения места любого элемента требуется только один индекс, размер массива равен n. Таблица $\begin{pmatrix} a_{11}, ..., a_{1n} \\ a_{m1}, ..., a_{mn} \end{pmatrix}$ – двумерный массив, размер массива равен $m \times n$.

Чтобы получить доступ к нужному элементу, нужно указать имя массива и комбинацию индекс. Индексы записывается через запятую (если размерность массива больше 1) в круглых скобках после имени массива. Если нужно присвоить, например, значение 5 третьему элементу массива A, нужно написать: A(3) = 5.

При выполнении этого оператора будет найден массив A, найдено соответствующее 3-му элементу место в памяти и в это место запишется число 5.

Индекс может быть переменной. Если нужно всем пяти элементам массива A присвоить значение 5, можно написать

```
FOR I = 1 TO 5
A(I) = 5
NEXT I
```

Имя массива образуется так же как имя переменной. Массивы определяются в операторе описания массивов DIM. В операторе DIM указываются имя массива и в круглых скобках границы изменения индексов.

Например, оператор

```
DIM A(1 TO  3), B(1 TO  4, 1 TO  5)
```

описывает два числовых массива: ***A*** состоящий из трех элементов и ***B*** содержащий 4 строки и 5 столбцов. Можно указать только верхние границы, в этом случае нижние границы будут равны нулю.

В соответствии с оператором DIM в памяти ЭВМ выделяется место для размещения этих массивов. Так, для массива ***A*** будут выделено место в памяти для трех элементов, для массива ***B*** – для двадцати элементов. Под

двумерный массив выделяется линейный участок памяти, в котором массив располагается по строкам.

При описании динамических массивов (количество элементов задается переменной, значение которой определяется до описания массива) используется оператор ReDIM.

Пример:

```
k = Cells(2, 1): m = Cells(2, 1)
ReDim a(1 To k), b(1 To k), C(1 To m), D(1 To m)
```

В VISUAL BASICe обработка массивов, а также ввод – вывод массивов осуществляется поэлементно.

Например, ввод одномерного массива ***A***, содержащего 11 элементов, расположенных в диапазоне ячеек Excel A1 ÷A11, можно осуществить при помощи операторов

```
dim a(10)
for i = 0 to 10
a(i)=cells(i,1)
next i
```

Ввод двумерного массива ***B***(4,5) , содержащего 20 элементов, расположенных в диапазоне ячеек Excel A1 ÷E4, можно осуществить при помощи операторов

```
dim b(1 то  4, 1 то  5)
for i = 1 to 4
for j = 1 to 5
b(i,j)=cells(i,j)
next j
next i
```

Ввод двумерного массива в приведенной программе осуществляется по строкам. Однако ввод этого массива можно осуществить и по столбцам:

```
dim b(1 то  4, 1 то  5)
for j = 1 to 5
for i = 1 to 4
b(i,j)=cells(i,j)
next i
next j
```

Вывод двумерных массивов в ячейки Excel можно осуществлять также, как по строкам, так и по столбцам, либо вообще в любом другом порядке.

Пример 9. Даны два массива x_i и y_i (i=1,10). Составить программу для вычисления

$$t = m\sum_{i=1}^{10}(x_i^2 - ny_i)$$

Программа для примера 9

```
sub pr9()
dim x(10), y(10)
m = cells(1, 2): n = cells(2, 2)
s=0
for i=1 to 10
x(i)=cells(3,i+1): y(i)=cells(4,i+1)
s=s+x(i)^2-n*y(i)
next i
t=m*s: cells(5,1)= " t= ": cells(5,2)= t
end sub
```

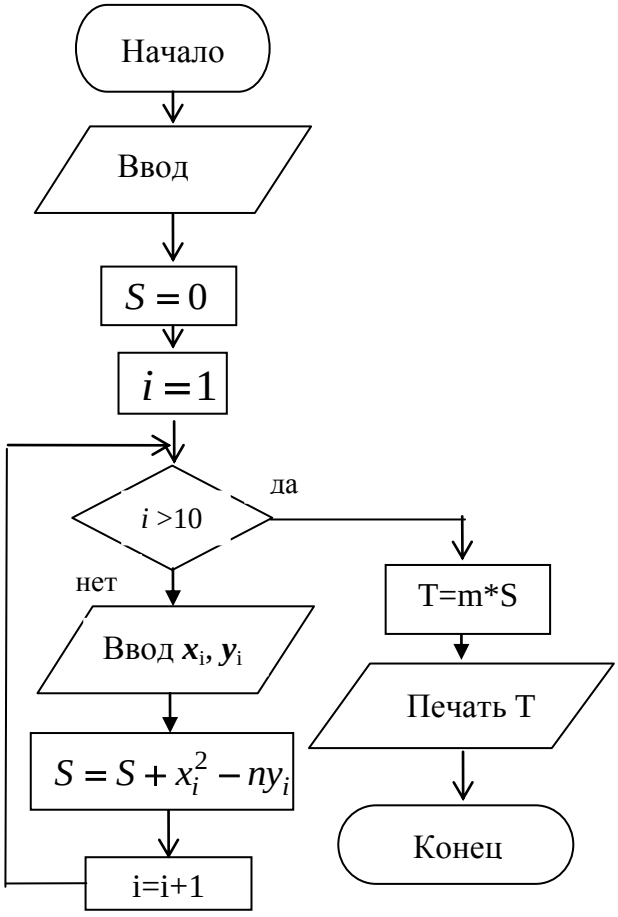


Рис. 2.6. Блок - схема программы на обработку массивов

Пример 10. Найти сумму элементов каждой строки массива ***C***(4,5).

```
Sub Pr10()
dim c(4,5), s(4)
for i = 1 to 4
s(i)=0
for j = 1 to 5
c(i,j)=cells(i, j): s(i) = s(i)+c(i,j)
next j
cells(i, j)=” ”: cells(i, j+1)= s(i)
next i
end sub
```

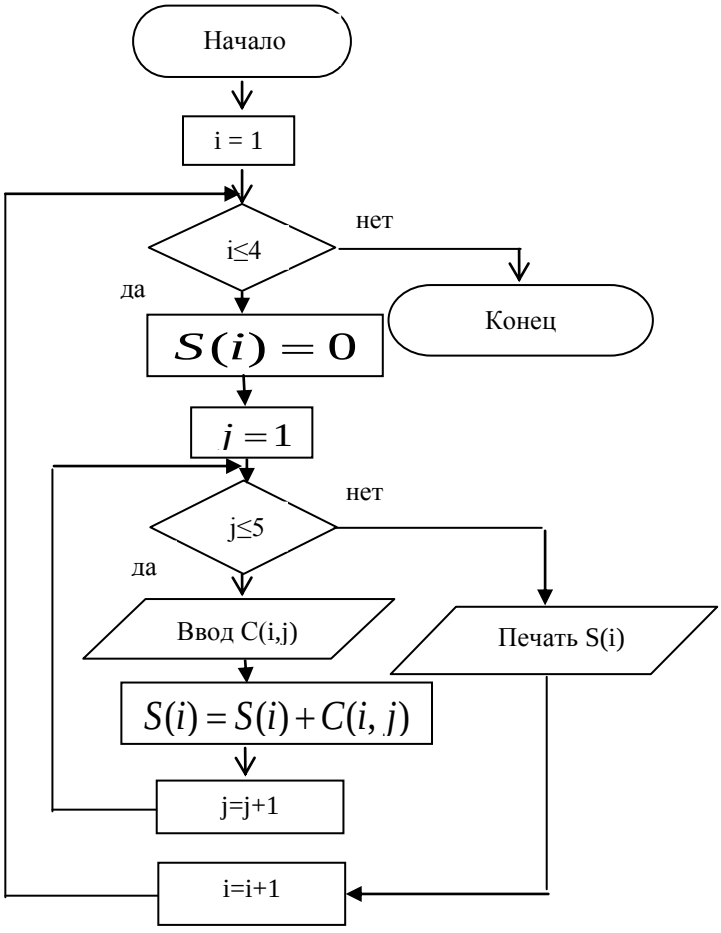
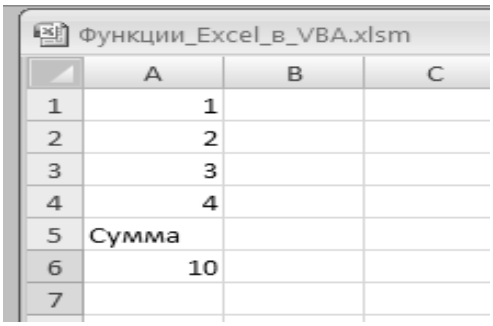


Рис. 2.7. Блок - схема циклического вычислительного процесса с применением двумерного массива

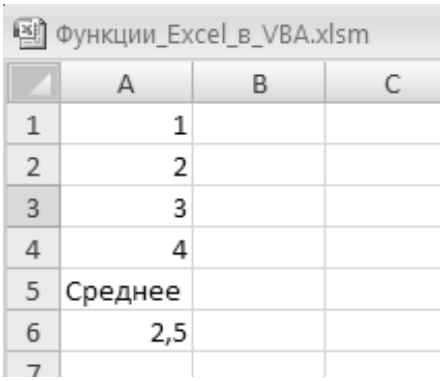
В программе, созданной в VBA, можно использовать встроенные функции Excel. Встроенные функции Excel являются ***свойствами*** объекта **Application.**

В следующих двух примерах встроенные функции Excel Sum (СУММ) и Average (СРЗНАЧ), вычисляющие сумму и среднее для чисел, расположенных в диапазоне A1:A4, являются ***свойствами*** объекта **Application.**

```
Sub Functii_Excel_v_VBA()
Dim A As Single
A = Application.
Sum(Worksheets("Лист1").
Range("A1:A4"))
Cells(5, 1) = "Сумма"
Cells(6, 1) = A
End Sub
```




```
Sub Functii_Excel_v_VBA()
Dim A As Single
A = Application.
Average(Worksheets("Лист1").
Range("A1:A4"))
Cells(5, 1) = "Среднее"
Cells(6, 1) = A
End Sub
```



Здесь также использовались объекты Worksheets (Листы) и Range (Диапазон).

Свойства объекта **Range** можно рассмотреть на следующих примерах:

Formula – формула в ячейках диапазона.

Операторы :

Range(“C2:C8”).Formula = “=$A5+sin($A$11)”

Range(“2:3”).Formula = “=$A5+sin($A$11)”

Range(“C3”).Formula = “=$A5+sin($A$11)”

Используются для ввода формулы “=$A5+sin($A$11)

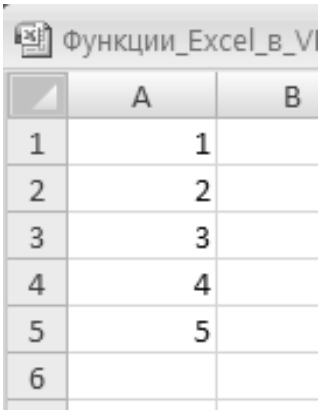
Value – массив значений в ячейках диапазона; **Columns** – семейство столбцов, из которых состоит диапазон; **Rows** - семейство строк, из которых состоит диапазон; **Cells** - семейство ячеек, из которых состоит диапазон.

Можно отметить у объекта еще два метода объекта **Range:**

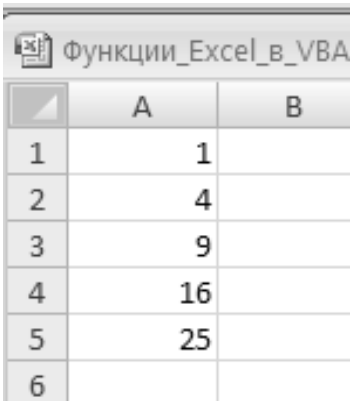
Select – выделение (активизация) ячеек диапазона; **Clear** - удаление содержимого ячеек диапазона.
Например, оператор **Worksheets("Лист1").Range("A1:A5").Clear** – очищает ячейки диапазона A1:A5.

В следующем примере объект **типа Range** используется в операторе цикла для возведения в квадрат значений ячеек диапазона A1:A5:

```
Sub kvadr()
Dim y As Range
For Each y In Worksheets("Лист1").Range("A1:A5")
y.Value = y.Value ^ 2
Next
End Sub
```



Исходные значения



Возведенные в квадрат

У объекта **Application** есть и другие свойства. Перечислим некоторые (наиболее часто применяющиеся) из них.

- **ActiveWorkbook** – активная книга.
- **ActiveSheet** – активный лист активной книги.
- **ActiveCell** – активная ячейка на активном листе активной книги.

В этой программе в активной ячейке устанавливается шрифт «*курсив*» и в нее записывается текст «сумма».

```
Sub Курсив()
With Application.ActiveCell.Font.Italic = True
Value = "Сумма"
End With
End Sub
```

ЛИТЕРАТУРА

1. Г. Зельднер «Программирование на языке Quick Basic 4 .Б» М.: ABF:1996 г.-186 с.
2. Майкл Маккелви. Visual Basic 4.-М.: Вином,1996.-23 с.
3. Методические указания к расчетно-графическим работам для студентов 1 курса специальностей 2903, 2906, 2907, 2908, 2909, 2910 / КГАСА, Сост.О.В.Бахарева. Казань, 1995, 34 с.
4. Методические указания по курсу "Информатика" для лабораторных и контрольных работ для студентов всех специальностей и направлений подготовки. Основы программирования. /Казанский государственный архитектурно-строительный университет; Сост.: Ф.Г.Ахмадиев, Ф.Г.Габбасов, - Казань, 2012. 44 с.
5. Г.З. Гарбер. «Основы программирования на Visual Basic и VBA в Excel 2007». Москва: «Солон – Пресс», 2008 г.- 191 с.,

ОСНОВЫ ПРОГРАММИРОВАНИЯ В VBA

Методические указания по курсу "Информатика" для лабораторных и контрольных работ для студентов всех специальностей и направлений подготовки.

Составители: Ахмадиев Фаил Габдулбарович,
Бекбулатов ИРек Гумарович,
Габбасов Фарит Гаязович

Редактор Е.А.Кириллович

Издательство
Казанского государственного архитектурно-строительного университета

Подписано в печать Формат 60x84/16
Заказ № Печать ризографическая Усл.печ.л.
Тираж Бумага офсетная №1 Уч. -изд.л.

Отпечатано в полиграфическом секторе
Издательства КГАСУ
420043, г. Казань, ул. Зеленая, 1

РЕЦЕНЗИЯ

Основы программирования в VBA
Ахмадиева Ф.Г., Бекбулатов И.Г., Габбасова Ф.Г.

Данные методические указания предназначены для лабораторных и самостоятельных работ для студентов всех специальностей и направлений подготовки и используются при выполнении контрольных работ заочниками по курсу "Информатика" на языке VBA**VISUAL BASIC**.

Считаю, что данные методические указания могут быть опубликованы.

Доктор физ.-мат. наук,
профессор, зав. кафедрой ВМ _____Р.Б.Салимов