

ЛАБОРАТОРНЫЕ РАБОТЫ

ПРОГРАММИРОВАНИЕ В VBA

ЛАБОРАТОРНАЯ РАБОТА №1

Изучение интерфейса редактора VBA.
Запись и чтение данных из ячеек Excel.
Диалоговые окна (InputBox\MsgBox).

ЛАБОРАТОРНАЯ РАБОТА №2

Vba. Создание собственных диалоговых окон

ЛАБОРАТОРНАЯ РАБОТА №3

Алгоритмы и программы разветвляющейся структуры

ЛАБОРАТОРНАЯ РАБОТА №4

VBA. Цикл с параметром (For...Next)
Массивы.

ЛАБОРАТОРНАЯ РАБОТА №5

Изучение и применение операторов цикла с предусловием и постусловием.

ЛАБОРАТОРНАЯ РАБОТА №6

VBA. Элемент управления Переключатель.

ЛАБОРАТОРНАЯ РАБОТА №7

VBA: Элемент управления Список.

ЛАБОРАТОРНАЯ РАБОТА №8

Подпрограммы и их применение

ЛАБОРАТОРНАЯ РАБОТА №1

VBA. ЭЛЕМЕНТЫ УПРАВЛЕНИЯ

ЦЕЛЬ РАБОТЫ: 1. Изучение интерфейса редактора VBA

2 Изучение объектов Visual Basic for Application на примере линейной программы

КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ:

VBA относится к языкам объектно- ориентированного программирования (ООП). ООП можно описать как методику анализа, проектирования и написания приложений с помощью объектов. *Объект* – комбинация кода и данных, которая может рассматриваться как единое целое, например элемент управления, форма и компонент приложения. Каждый объект определяется по принадлежности к классу. Все визуальные объекты, такие как рабочий лист (Worksheet), диапазон (Range), диаграмма (Chart), форма (UserForm), являются объектами.

Доступ к данному языку программирования можно осуществлять практически из любого приложения Windows. Мы будем работать вместе с Microsoft Excel, который будет являться основным приложением для проекта VBA.

Редактор VBA активизируется командой **Сервис, Макрос, Редактор Visual**

Basic. Возвратиться из редактора VBA в рабочую книгу можно нажатием кнопки **Вид Microsoft Excel.** Интерфейс редактора VBA состоит из следующих основных компонентов:

- окно проекта,
- окно свойств,
- окно модуля (окно редактирования кода),
- окна форм,
- меню и панели инструментов (рисунок 1).

Окно проекта

Окно проекта в редакторе VBA активизируется выбором команды **Вид, окно проекта** или нажатием кнопки **Окно проекта**



В окне проекта представлена иерархическая структура файлов форм и модулей текущего проекта

В проекте автоматически создается модуль для каждого рабочего листа и для всей книги. Кроме того, модули создаются для каждой пользовательской формы макросов и классов. По своему назначению модули делятся на два типа- *модули объектов* и *стандартные*. К стандартным модулям относятся те, которые содержат макросы. Такие модули добавляются в проект командой **Вставка Модуль**. К модулям объектов относятся модули, связанные с рабочей книгой, рабочими листами, формами, и модули класса.

Окно свойств

Окно свойств

В окне свойств перечисляются основные установки свойств выбранной формы или элемента управления. Используя это окно, можно просматривать свойства и изменять их установки. Для просмотра свойств выбранного объекта надо либо щелкнуть кнопку **Окно свойств**, либо выбрать команду **Вид, Окно свойств**.

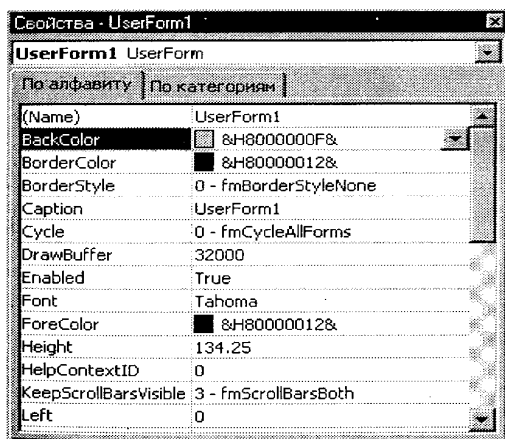


Рис. 1.9. Окно свойств

Окно свойств состоит из двух составных частей: верхней и рабочей. В верхней части окна свойств располагается раскрывающийся список, из которого можно выбрать любой элемент управления текущей формы или саму форму. Рабочая часть состоит из двух вкладок: **По алфавиту** (Alphabetic) и **По категориям** (Categorized), отображающие набор свойств в алфавитном порядке или по категориям. В обеих вкладках свойство Name (имя элемента управления) будет первым. Изменяются значения свойств одним из следующих способов:

- Вводом с клавиатуры значения свойства в соответствующее поле.
- Значения большинства свойств можно выбрать из раскрывающегося списка.

Раскрывающийся список активизируется щелчком в соответствующем поле окна свойств.

Окно модуля (окно редактирования кода)

Программа (код программы) записывается в окне кода (окно модуля). Окно кода используется при написании любой программы VBA, будь это код макроса, запуск которого осуществляется при нажатии кнопки в созданной пользователем форме, или подпрограмма. Код программы вводится непосредственно в окно кода, так же как текст в любом текстовом редакторе.

Для того чтобы получить окно модуля, необходимо выполнить следующие действия: Вставка / Модуль (Insert / Module).

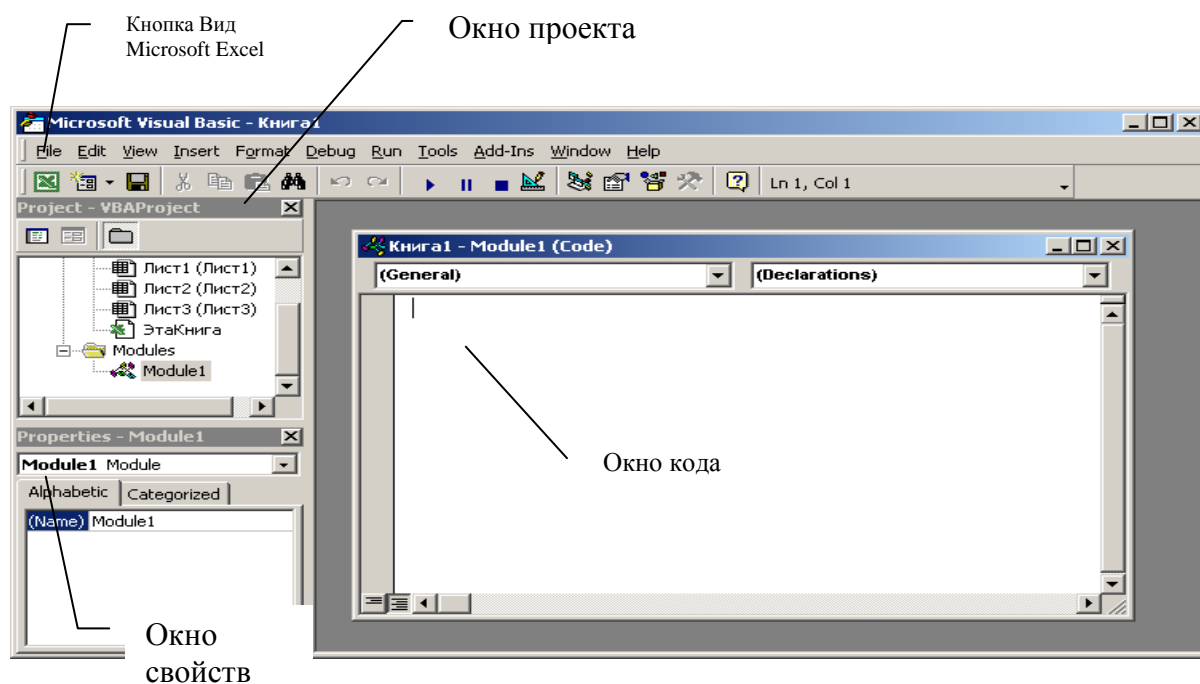


Рисунок 1 - Интерфейс редактора VBA

Объект обладает определенными свойствами и методами.

Свойства - это характеристики объекта, такие как размер, цвет, положение на экране, или состояние объекта, например доступность или видимость.

Методы – это действия, выполняемые над объектом.

Пример.

```
Worksheets("Лист1").Visible = False
```

```
Worksheets ("Лист 2").Delete
```

С помощью установки свойству

Видимость значения Ложь

скрывается рабочий лист “Лист 1”.

При помощи метода Delete удаляется этот рабочий лист “Лист 2” .

Программный объект может являться частью другого, большего программного объекта. Для доступа к свойствам и методам объекта, являющегося составной частью более крупного объекта, нужно определить каждый из сборных объектов, начиная с самого левого (большого объекта), а затем поставить точку и указать следующий, составной объект, затем опять ставится точка и определяется следующий внутренний объект, пока не будет определен объект, к свойствам и методам которого необходимо получить доступ.

Пример.

```
Workbooks ("Книга1") .Worksheets ("Лист1") .Range ("A14") .Font
```

 – получаем доступ к рабочей книге "Книга1", рабочему листу “Лист1”, шрифту ячейки "A14".

Объектами Excel являются таблицы, рабочие книги, диаграммы, области ячеек и др. Семейство представляет собой объект, содержащий несколько других объектов, как правило, одного и того же типа.

Например, семейство Workbooks объединяет все открытые рабочие книги.

Обратиться к элементу семейства можно по имени или номеру.

Пример:

```
Worksheets ("Лист1") или Worksheets (1)
```

Переменные - это поименованные области в памяти компьютера. После вычисления какого-либо значения оно записывается в память, чтобы затем можно было к нему возвращаться. Использование переменных дает VBA возможность создавать прямое соответствие между областями памяти и заданным именем. Затем можно использовать это имя в программе.

Имя переменной может содержать цифры, буквы и знак подчеркивания, но обязательно должно начинаться с буквы.

Константы, в отличие от переменных, не могут изменять свои значения. Использование констант делает программы легче читаемыми и позволяет проще вносить исправления — отпадает необходимость многократно исправлять значения по тексту программы, т. к. достаточно ввести новое значение при определении константы.

– **Const <ИмяКонстанты> [As Single] = <Выражение>**

– **Пример**

```
Const ПроцентнаяСтавка As Single =0.2
```

```
Const g=9.8
```

Над переменными и константами могут выполняться операции.

В таблице 3 представлены математические операции VBA.

Таблица 3

Выражение	Операция	Пример		
		A	B	Результат
A+B	Сложение	5	2,75	7,75
A – B	Вычитание	5	2,75	2,15
A * B	Умножение	2	6	12
A/B	Деление	7	2	3.5
A \ B	Целочисленное деление	7	2	3
A mod B	Остаток от деления по модулю	7	2	1
A^B	Возведение в степень	2	3	8

Стандартные математические функции VBA представлены в таблице

Обращение	Функция
Abs (x)	Модуль аргумента
Atn(x)	Арктангенс (радианы)
Cos (x)	Косинус (x в радианах)
Exp (x)	e ^x — экспонента
Int (x)	Целая часть x, полученная отбрасыванием дробной части
Fix(x)	Число, округленное до ближайшего меньшего целого
Log(x)	Натуральный логарифм
Sin(x)	Синус (x—в радианах)
Sqr(x)	Корень квадратный
Tan(x)	Тангенс числа

Старшинство операций (в порядке убывания приоритета):

- ⇒ операции в скобках;
- ⇒ вычисление функции;
- ⇒ ^ ;
- ⇒ смена знака;
- ⇒ *, /, \, mod;
- ⇒ +, -
- ⇒ =, >, <, >=, <=, <> ,
- ⇒ Not,
- ⇒ And,
- ⇒ Or,
- ⇒ Xor.

Логические выражения в результате вычисления принимают логические значения True (Истина) или False (Ложь). Операндами логического выражения могут быть логические константы, переменные логического типа, отношения. В VBA чаще используют 4 логические операции: отрицание — NOT, логическое умножение — AND, логическое сложение — OR, исключающее “или” — XOR. Результаты логических операций для различных значений операндов приведены в таблице 5. Использованы обозначения: T — True, F — False.

Таблица 5

A	B	not A	A and B	A or B	A xor B
T	T	F	T	T	F
T	F	F	F	T	T

F	F	T	F	F	F
F	T	T	F	T	T

Окна форм,

Для создания диалоговых окон, разрабатываемых приложений в VBA, используются формы. Редактор форм является одним из основных инструментов визуального программирования. Форма в проект добавляется с помощью команды **Вставка, Форма** (Insert, Form) или нажатием кнопки **Вставить UserForm**

В результате на экран выводится незаполненная форма с панелью инструментов **Панель элементов** (рис. 1).

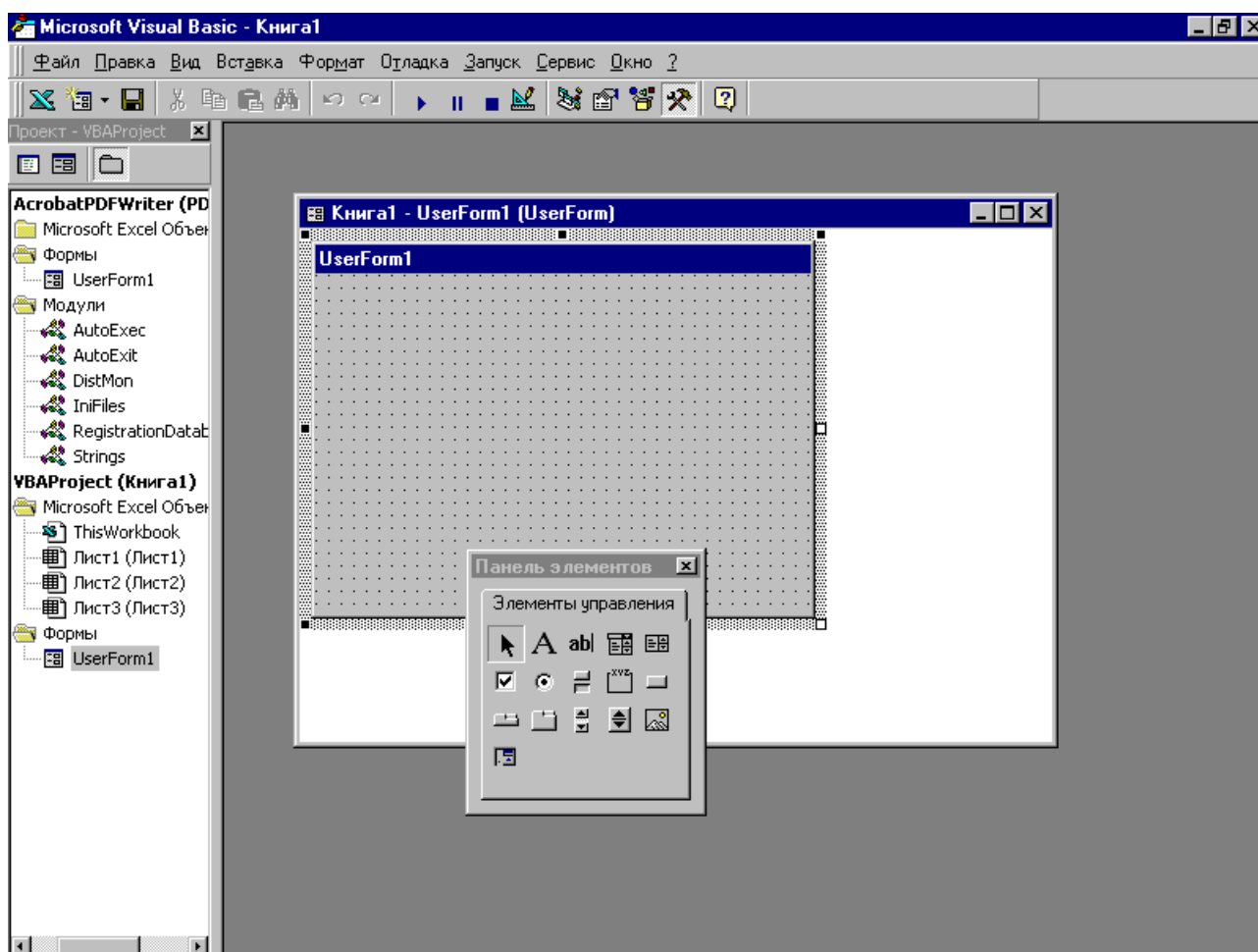


Рисунок 1

Используя панель инструментов **Панель элементов** из незаполненной формы, можно сконструировать любое требуемое для приложения диалоговое окно. Размещение нового управляющего элемента в форме осуществляется следующей последовательностью действий:

1. Щелкните значок того элемента, который вы собираетесь разместить в форме.
2. Поместите указатель мыши на то место, где будет располагаться управляющий элемент.
3. Нажмите левую кнопку мыши и, не отпуская ее, растяните появившийся прямоугольник до требуемых размеров.
4. Отпустите кнопку мыши. Элемент управления на нужном месте создан.

Размеры формы и расположенных на ней элементов управления можно изменять. Технология изменения размеров стандартная для Windows: выделить изменяемый элемент, разместить

указатель мыши на одном из размерных маркеров и протащить его при нажатой левой кнопки мыши так, чтобы объект принял требуемые размеры. Окно редактирования форм поддерживает операции буфера обмена. Таким образом, можно копировать, вырезать и вставлять элементы управления, расположенные на поверхности формы.

Любой управляющий элемент, который вы разместили на форме, обладает рядом свойств

ОПЕРАТОР ПРИСВОЕНИЯ

Оператор присвоения присваивает значение выражения переменной, константе или свойству объекта. Оператор присвоения всегда включает знак равенства (=).

Синтаксис:

– Переменная (или Постоянная) = Выражение

Оператор присвоения предписывает выполнить выражение, заданное в его правой части, и присвоить результат переменной, имя которой указано в левой части.

Например:

$x = 2$

$x = x + 2$

в результате выполнения этих команд переменной x будет присвоено значение 4.

Организация ввода/вывода

Решение любой задачи имеет три части:

1. Ввод данных
2. Обработка данных
3. Вывод результата

Под вводом данных понимается описание всех переменных, констант и массивов, используемых в программе, а также код, обеспечивающий присвоение этим переменным вводимых данных.

Под так называемой обработкой данных понимается код, состоящий из математических выражений, которые приводят к получению результата.

Вывод результата – это код программы, который позволяет отобразить полученный результат в необходимом виде: на экране (лист excel, форма), на принтере и т.д.

Ввод/вывод данных можно осуществлять несколькими способами:

- запись и чтение данных из ячеек Excel.
- диалоговые окна (InputBox\MsgBox).
- конструирование собственных диалоговых окон (UserForm)

Сохранение программы

- 1) Если программу сохраняется первый раз или вас устраивает уже существующее имя, то сохранять можно как в VBA, так и в Excel;
- 2) Если вас не устраивает уже существующее имя, то сохранять необходимо таким образом: выйти в Excel, выбрать пункт меню *Файл*, а в нем пункт - *Сохранить как*.

РАССМОТРИМ ПЕРВЫЙ ИЗ НИХ.

Производить запись и чтение данных из ячеек Excel можно используя два свойства: Range и Cells.

Свойство Range требует один аргумент – имя ячейки, записанное при помощи относительной или абсолютной адресации.

Например:

Range("A1")

Range("\$A\$1")

Свойству Cells требуется два аргумента, номер строки и номер столбца, на пересечении которых находится ячейка.

Например:

Cells(2,4) (обращение к ячейке D2)

ПОРЯДОК ВЫПОЛНЕНИЯ:

1. Выполнить команду СЕРВИС/МАКРОС/ РЕДАКТОР VBA
2. Выполнить команду ВСТАВКА/МОДУЛЬ (Insert/Module)
3. Выполнить команду ВСТАВКА/Процедура (Insert/Procedure)
4. Ввести имя процедуры (по правилу именования)

Набрать программу для вычисления выражения $c = \sin(a) / \cos(b)$.

Текст программы:

Public Sub ggg() ' заголовок процедуры

Dim a As Double, b As Double, c As Double 'раздел описания переменных

'исполняемая часть программы

a = Worksheets(1).Range("a1").Value 'присваивание идентификатору a числового значения
' ячейки a1 рабочего листа

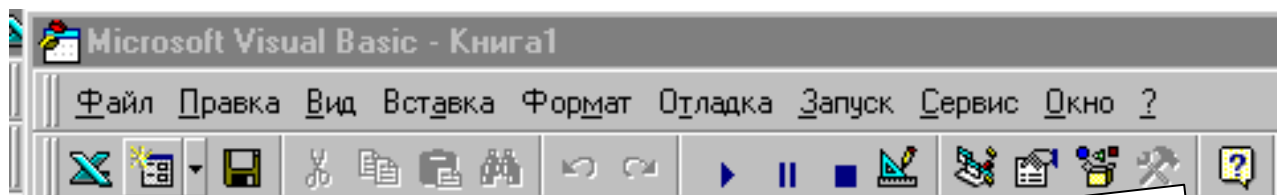
b = Worksheets(1).Range("b1").Value ' присваивание идентификатору b числового значения
' ячейки b1 рабочего листа

c = Sin(a) / Cos(b) ' вычисление арифметического выражения

Worksheets(1).Range("c1").Value = c ' передача вычисленного значения в ячейку c1 рабочего
' листа 1

End Sub 'конец процедуры

5. Запустить программу на выполнение. Запуск осуществляется или нажатием клавиши F5, или кнопка на панели инструментов.



6. Просмотреть результаты выполнения на рабочем листе.

Схема алгоритма.



РАССМОТРИМ ВТОРОЙ СПОСОБ ОРГАНИЗАЦИИ ВВОДА/ВЫВОДА.

Для организации ввода/вывода также можно использовать диалоговые окна. Наиболее часто в программах VBA встречаются две разновидности диалоговых окон: окна сообщений и окна ввода

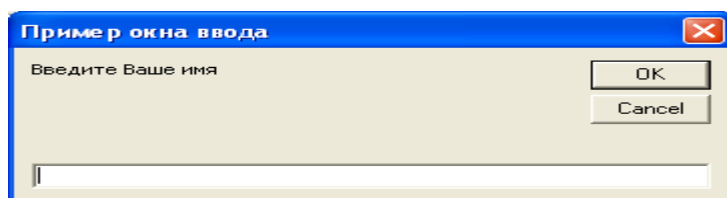
Окно сообщения используется для предоставления информации пользователю. Можно задать вывод на экран окна сообщения, в котором пользователь должен щелкнуть на одной из кнопок, прежде чем продолжить работу.

Окно ввода создается и выводится на экран с помощью функции InputBox.

Функция InputBox	<p>Выводит на экран диалоговое окно, содержащее сообщение, поле ввода и две кнопки OK и Cancel. Устанавливает режим ожидания ввода текста пользователем или нажатия кнопки, а затем возвращает значение типа string по нажатию кнопки OK, содержащее текст, введенный в поле. При нажатии кнопки Cancel возвращает пустую строку.</p> <p>Синтаксис:</p> <pre>InputBox(prompt[, title] [, default])</pre> <ul style="list-style-type: none"> – prompt — строковое выражение, отображаемое как сообщение в диалоговом окне. Строковое значение prompt может содержать несколько строк. Для разделения строк допускается использование символа возврата каретки (chr(13)), символа перевода строки (chr(10)) или комбинацию этих символов (chr(13) & Chr(10)); – title — строковое выражение, отображаемое в строке заголовка диалогового окна. Если этот аргумент опущен, в строку заголовка
---------------------	--

Например:

- Имя=InputBox(“Введите Ваше имя”, “Пример окна ввода”)
- На экране появится окно.



Переменной Имя будет присвоено значение типа String, введенное пользователем.

Следует учесть, что, поскольку введенные пользователем данные считаются текстом, при вводе числовых значений необходимо преобразовать их к одному из числовых типов данных с помощью функции преобразования типа, например CDb1.

– `X=CDbl (InputBox ("Введите значение X", "Пример окна ввода", "1,678"))`

Введенное пользователем значение будет преобразовано к типу Double и присвоено переменной X. Если пользователь не будет вводить значение, а просто нажмет кнопку OK, переменной X будет присвоено значение по умолчанию – 1.678.

Процедура MsgBox	<p>Выводит на экран диалоговое окно, содержащее сообщение, устанавливает режим ожидания нажатия кнопки пользователем, а затем возвращает значение типа integer, указывающее, какая кнопка была нажата. Синтаксис:</p> <p><code>MsgBox(prompt[, buttons] [, title])</code></p> <p>Аргументы:</p> <p><code>prompt</code> — строковое выражение, отображаемое как сообщение в диалоговом окне;</p> <p><code>buttons</code> — числовое выражение, представляющее сумму значений, которые указывают <u>число и тип отображаемых кнопок</u>, тип используемого значка, основную кнопку</p> <p><u>Значения констант определяющих число и тип кнопок используемого значка</u></p>
---------------------	---

Значения параметра Buttons процедуры MsgBox, определяющие отображаемые в диалоговом окне кнопки, приведены в таблице 11:

– **Таблица 11**

Константа	Значение	Отображаются кнопки
<code>vbOKOnly</code>	0	OK
<code>VbOKCancel</code>	1	OK, Отмена
<code>VbAbortRetryIgnore</code>	2	Стоп, Повтор, Пропустить
<code>VbYesNoCancel</code>	3	Да, Нет, Отмена
<code>VbYesNo</code>	4	Да, Нет
<code>VbRetryCancel</code>	5	Повтор, Отмена

При написании программ с откликом, когда нужно знать, какая кнопка диалогового окна была нажата (таблица 12), вместо возвращаемых значений удобнее использовать следующие константы VBA, которые делают код программы более читаемым и, к тому же, их легко запомнить.

Таблица 12

Константа	Значение	Нажатая кнопка
vbOK	1	ОК
vbCancel	2	Отмена (Cancel)
vbAbort	3	Прервать (Abort)
vbRetry	4	Повторить (Retry) Пропустить (Ignore)
vbIgnore	5	Да (Yes)
vbYes	6	Нет (No)
vbNo	7	

Пример.

- **N = MsgBox ("Значение переменной X=" & X & Chr(10) & "Продолжить вычисления?", VbYesNo, "Пример окна MsgBox")**
- Если к моменту выполнения данного оператора переменная X равнялась числу 2,14587895, то на экране появится следующее окно

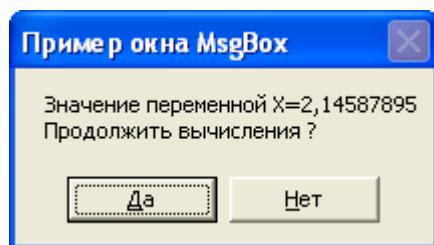
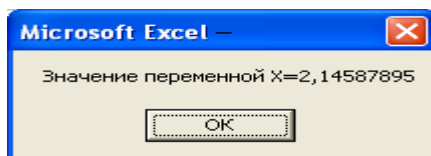


Рисунок 4 – Пример окна сообщений

Пользователь может нажать одну из кнопок – Да или Нет. Если будет нажата кнопка Да, переменной N будет присвоено значение 6, если будет нажата кнопка Нет – 7. Проанализировав в дальнейшем это значение, можно выбрать одну из ветвей выполнения программы.

- **Часто процедура MsgBox используется в «минимальном» варианте - только для вывода сообщения, с одной кнопкой – ОК. В этом случае аргументы не берутся в скобки.**
- Например:

- **MsgBox "Значение переменной X=" & X**



кнопку





- Рисунок 5 – Пример окна сообщений
- Значения параметра Buttons процедуры MsgBox, определяющие основную

Константа	Значение	Описание
vbDefaultButton1	0	First button is default (default)
vbDefaultButton2	256	Second button is default
vbDefaultButton3	512	Third button is default
vbDefaultButton4	768	Fourth button is default

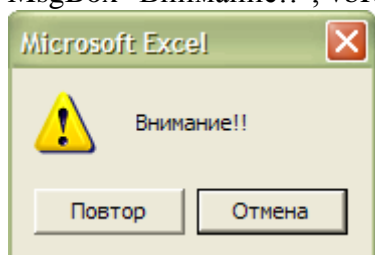
приведены в таблице:

- Значения параметра Buttons процедуры MsgBox, определяющие отображаемые

в диалоговом окне используемые информационные значки

Константа	Значение	Описание	Вид значка
vbCritical	16	Критическое сообщение	
vbQuestion	32	Предупреждение запроса	
vbExclamation	48	Предупреждающее сообщение	
vbInformation	64	Информационное сообщение	

-
- MsgBox "Внимание!!", 5 + 256 + 48
- MsgBox "Внимание!!", vbRetryCancel + vbDefaultButton2 + vbExclamation



Пример окна сообщений

Набрать программу для вычисления выражения $c = \sin(a) / \cos(b)$.



```
Public Sub www()  
Dim a As Double, b As Double, c As Double  
a = CDbl(InputBox("Введите a", "Задача", 2.4))  
b = CDbl(InputBox("Введите b", "Задача", 1.4))  
c = Sin(a) / Cos(b)  
MsgBox "Значение выражения  $c = \sin(a) / \cos(b)$  получилось=" & c, vbInformation, "Ответ"  
End Sub
```

ЛАБОРАТОРНАЯ РАБОТА N 2

VBA. ЛИНЕЙНАЯ ПРОГРАММА НА ОСНОВЕ СОЗДАНИЯ СОБСТВЕННЫХ ДИАЛоговых ОКОН

По своей сути форма (или пользовательская форма) представляет собой диалоговое окно, в котором можно размещать различные элементы управления. В приложении может быть как одна, так и несколько форм. Новая форма добавляется в проект выбором команды **Вставка (Insert) → UserForm**.

В VBA имеется обширный набор встроенных элементов управления. Используя этот набор и редактор форм, нетрудно создать любой пользовательский интерфейс, который будет удовлетворять всем требованиям, предъявляемым к интерфейсу в среде Windows. Элементы управления являются объектами. Как любые объекты, они обладают свойствами, методами и событиями. Элементы управления создаются при помощи Панели элементов, которая отображается на экране либо выбором команды **Вид (View) → Панель элементов (Toolbox)**,

либо нажатием кнопки  панели инструментов **Standard**. На этой панели представлены кнопки, позволяющие конструировать элементы управления. Для создания элементов управления служат все кнопки панели инструментов, за исключением кнопки **Выбор объекта** . Щелкнув по кнопке **Выбор объекта**, можно выбрать уже созданный в форме элемент управления для последующего его редактирования (изменения размеров или редактирования).

Приводим список основных элементов управления и соответствующих кнопок панели элементов.

Таблица 16

Элемент управления	Имя	Кнопка, его создающая	Элемент управления	Имя	Кнопка, его создающая
Поле	TextBox		Переключатель	OptionButton	
Надпись	Label		Флажок	CheckBox	
Кнопка	CommandButton		Выключатель	ToggleButton	
Список	ListBox		Рамка	Frame	
Поле со списком	ComboBox		Рисунок	Image	
Полоса прокрутки	ScrolBar		Набор страниц	MultiPage	
Счетчик	SpinButton		Набор вкладок	TabStrip	

Для размещения элемента управления на лист или в форму необходимо нажать соответствующую кнопку на панели элементов и с помощью мыши перетащить рамку элемента управления в нужное место. После этого элемент управления можно перемещать, изменять его размеры, копировать в буфер обмена, вставлять из буфера обмена и удалять из формы.

Приводим основные общие свойства элементов управления.

Таблица 17

Свойство	Описание
Caption	Надпись, отображаемая при элементе управления
AutoSize	Допустимые значения: True (устанавливает режим автоматического изменения размеров элемента управления так, чтобы на нем полностью помещался текст, присвоенный свойству Caption) и False (в
Visible	Допустимые значения: True (элемент управления отображается во время выполнения программы) и False (в противном случае)
Enabled	Допустимые значения: True (пользователь вручную может управлять элементом управления) и False (в противном случае)
Height и Width	Устанавливают геометрические размеры объекта (высоту и ширину)
Left и Top	Устанавливают координаты верхнего левого угла элемента управления, определяющие его местоположение в форме
ControlTipText	Устанавливает текст в окне всплывающей подсказки, связанной с элементом управления. В следующем примере элементу управления CommandButton назначен текст, всплывающей подсказки это кнопка:
BackColor, ForeColor и BorderColor	Устанавливают цвет заднего и переднего плана элемента управления, также его границы
BackStyle	Устанавливает тип заднего фона
BorderStyle	Устанавливает тип границы. Допустимые значения: fmBorderStyleSingle (граница в виде контура); fmBorderStyleNone (граница невидима)
SpecialEffect	Устанавливает тип границы. Отличается от свойства BorderStyle тем, что позволяет установить несколько типов, но одного цвета. BorderStyle позволяет установить только один тип, но различных цветов
Picture (создание картинки)	Внедряет картинку на элемент управления. Например, на поверхности кнопки картинка отображается с помощью следующей инструкции: CommandButton1. _ Picture =LoadPicture("c:\my doc\Круг.bmp") Функция LoadPicture (ПолноеИмяФайла) считывает графическое изображение. Аргумент ПолноеИмяФайла указывает полное имя графического файла
Picture (удаление картинки)	После того как картинка создана на элементе управления, иногда возникает необходимость ее удалить. Это легко достигается присвоением свойству Picture значения LoadPicture(" ") CommandButton1.Picture = LoadPicture(" ")

После размещения элементов управления на форме необходимо связать объект на форме с кодом.

В VBA очень просто связать объект с кодом. Для выполнения данной операции:

1. Дважды щелкните по элементу управления в форме. Появляется окно модуля для выбранного объекта. Выберите событие для которого требуется создать процедуру обработки, в списке, расположенном в верхнем правом углу окна модуля. Введите текст процедуры.
2. Вызвать контекстное меню необходимого объекта правой клавишей мыши и нажать поле *Программа*.

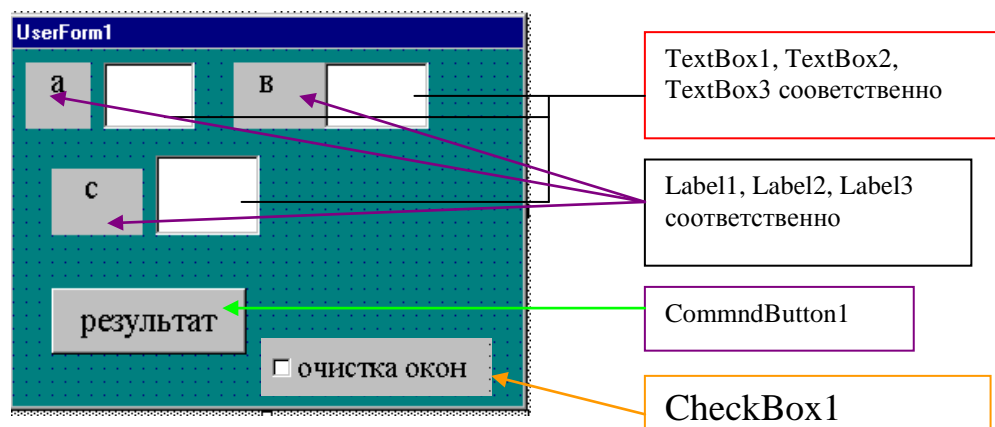
ЗАДАНИЕ К ВЫПОЛНЕНИЮ

1. Расположите на форме следующие элементы:
 - Label1;
 - TextBox1;
 - CommandButton1.
2. Активизируйте Label1, в окне свойств найдите свойство Caption и измените Label1 на название вашего факультета.
3. Те же действия произведите с CommandButton1, изменяя Caption на - "ок".
4. Активизируйте TextBox1 и измените свойство Text, набрав аббревиатуру своей группы.
5. С помощью элемента *Выбор объекта* выделите все элементы на форме. В окне свойств хорошо видно какие свойства одинаковы для всех элементов. Изменяя свойства Font (полужирный, курсив, размер шрифта15), BackColor (светлая тень для кнопки), Visible(False), проследите все изменения элементов управления на форме.
6. Осуществить запуск программы с помощью кнопки **Запуск** на панели инструментов или меню **Запуск/ Запуск программы**.
7. Вернуться в режим конструктора VBA для этого нажмите крестик на форме.
8. Выделите опять все объекты и поменяйте только свойство Visible (True) и снова произведите запуск программы.
9. Самостоятельно изменяйте другие свойства элементов данной формы и наблюдайте их изменения.

Решим задачу: найдем сумму $c = \sin(a)/\cos(b)$

Порядок выполнения работы:

1. Выполнить команду СЕРВИС /МАКРОС/РЕДАКТОР VBA
2. Выполнить команду ВСТАВКА/USER FORM
3. Поместить на форму элементы, требуемые для решения задачи, с панели элементов, и расположить их нужным образом.



4. Изменить свойства объектов на форме с помощью окна свойств.

Свойство	Значение
----------	----------

Label1.Caption	A
Label2.Caption	B
Label3.Caption	C
CommandButton1	Результат
CheckBox1.Caption	Очистка окон
Для всех объектов свойство .BackColor	По своему вкусу выбрать цвет Из палитры цветов
Для Label1, Label2 ,Label3 Свойство Font	В диалоговом окне “Шрифт”, которое появится после щелчка по Кнопке с изображением трех маленьких точек, расположенной напротив свойства Font в окне свойств, выбрать размер 16

5. Написать программный код. Для этого рекомендуется выполнить двойной щелчок по кнопке *результат* и перейти в окно программы, где набрать текст процедуры обработки события Click() для кнопки и для флажка(CheckBox1):

```
Private Sub CheckBox1_Click()
    TextBox1.Text = ""
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox3.Visible = False
    TextBox1.SetFocus
    CheckBox1.Value = False
End Sub

Private Sub CommandButton1_Click()
    Dim a As Integer
    Dim b As Integer
    Dim c As Integer
    a = CInt(TextBox1.Text)
    b = CInt(TextBox2.Text)
    c = a + b
    MsgBox "результат смотри в TextBox3"
    TextBox3.Visible = True
    TextBox3.Text = c
End Sub
```

Пояснения к программе:

1) **Dim a As Integer**

Эта инструкция описывает переменные как Integer — целые числа от -32768 и до 32767. При попытке присвоить а число, выходящее за пределы этого диапазона, возникает ошибка. При присваивании а дробного числа, выполняется округление.

Инструкция Dim - Описывает переменные и выделяет для них память.

2) **CInt** - функция преобразования типов данных (преобразовывает выражение в скобках к типу Integer).

Синтаксис **CInt(выражение)**

3) **c=a+b**

Оператор присваивания (=)-вычисляется значение выражения, стоящего справа от знака присваивания, и присваивается переменной, стоящей слева от знака присваивания.

4) **MsgBox "результат смотри в TextBox3"**

Появляется на экране окно сообщений MsgBox, в котором отображается сообщение, записанное в кавычках, и выполнение программы останавливается до тех пор пока не будет нажата кнопка "ОК".

5) **TextBox3.Text = c**

Результат выполнения программы (c) выводится на форму в TextBox3

6) **TextBox1.Text = ""**, **TextBox2.Text = ""**, **TextBox3.Text = ""**

Производится очистка полей TextBox1, TextBox2, TextBox3.

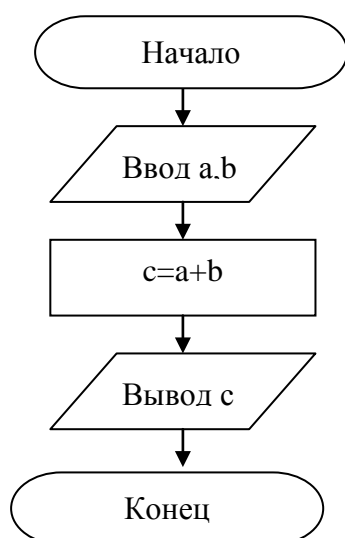
7) **TextBox1.SetFocus**

Устанавливается фокус (курсор) в TextBox1.

8) **CheckBox1.Value = False**

Исчезает галочка у флажка CheckBox1.

Блок-схема к программе



ЗАДАНИЯ ДЛЯ ВЫПОЛНЕНИЯ

В предложенных задачах ввод и вывод осуществить рассмотренными способами.

1.	$\alpha = \lg a^7 + \arctg x^2 + \frac{\pi}{\sqrt[3]{ a+x }}.$	$\gamma = 5 \cdot a^{nx} + \beta^{3/2} - \sqrt[3]{ \cos \alpha^4 }.$
2.	$\beta = 18 \cdot ax^2 + \sqrt[5]{y^2} + \sin \frac{\alpha}{2}.$	$x = \sin \alpha \cdot \cos \frac{\beta}{2} + \sqrt[5]{\alpha^2 + \beta^2}.$
3.	$y = \frac{\sin^2 \alpha + \operatorname{tg} \gamma}{\omega + \cos \alpha}.$	$z = \left(\sqrt{\frac{ax+b}{c+dx}} + \operatorname{tg} x \right)^{2/3} - e^{2x}.$
4.	$x = \frac{a\sqrt{\sin(\omega t + \varepsilon)} - e^{-\alpha t}}{\sqrt[3]{\ln(2k+d)} + d^{3k}}.$	$y = \frac{(\arctg x^3 + \cos \sqrt[3]{x})^{2x}}{e^x + \ln 2.4x^2 }.$
5.	$z = \left(\frac{ay}{a+b} + \frac{c}{ax^2 + bx} \right)^3 + \sin \omega t.$	$t = \frac{3x^2 + 25e^{x^2}}{ x^3 - 1 + \sqrt{ax^2 + \alpha}} + \ln^2 x.$

6.	$u = \operatorname{arctg} \alpha + \frac{\sin \beta^2}{2} - e^{\sqrt{ 3x }}.$	$v = e^x + a^3 \left(\frac{b}{a+b} \right)^{2/3} - 2 \sin x^2.$
7.	<p>Даны x, y, z. Вычислить a, b, если</p> $a = \frac{\sqrt{ x-1 } - \sqrt[3]{ y }}{1 + \frac{x^2}{2} + \frac{y^2}{4}},$ $b = x \left(\operatorname{arctg} z + e^{-(x+3)} \right)$	<p>Даны x, y, z. Вычислить a, b, если</p> $a = \frac{3 + e^{y-1}}{1 + x^2 y - \operatorname{tg} z },$ $b = 1 + y - x + \frac{(y - x)^2}{2} + \frac{ y - x ^3}{3}.$
8.	<p>Даны x, y, z. Вычислить a, b, если</p> $a = (1 + y) \frac{x + y / (x^2 + 4)}{e^{-x-2} + 1 / (x^2 + 4)},$ $b = \frac{1 + \cos(y - 2)}{x^4 / 2 + \sin^2 z}$	<p>Даны x, y, z. Вычислить a, b, если</p> $a = y + \frac{x}{y^2 + \left \frac{x^2}{y + x^3 / 3} \right },$ $b = \left(1 + \operatorname{tg}^2 \frac{z}{2} \right)^2$
9.	<p>Даны x, y, z. Вычислить a, b, если</p> $a = \frac{2 \cos(x - \pi / 6)}{1 / 2 + \sin^2 y},$ $b = 1 + \frac{z^2}{3 + z^2 / 5}.$	<p>Даны x, y, z. Вычислить a, b, если</p> $a = \frac{1 + \sin^2(x + y)}{2 + \left x - 2x / (1 + x^2 y^2) \right },$ $b = \cos^2 \left(\operatorname{arctg} \left(\frac{1}{z} \right) \right).$

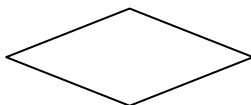
Лабораторная работа N3

Алгоритмы и программы разветвляющейся структуры

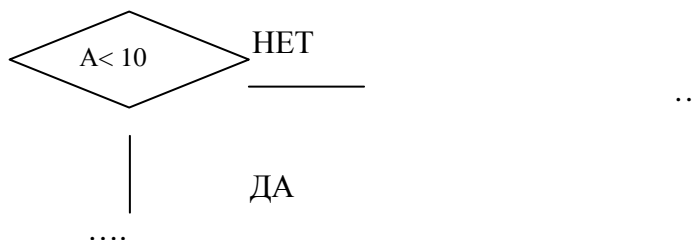
Цель работы: научиться разрабатывать алгоритмы и программы разветвляющейся структуры с условным оператором IF. Познакомиться с некоторыми объектами VBA и с их свойствами

КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ:

1. В схемах алгоритмов выбор условия обозначается с помощью символа,



от которого отходит ветвление для указания дальнейших действий в зависимости от выбора “ДА” или “НЕТ”. Выбор хода алгоритма, как правило, зависит от значения переменной или выражения, состояния объекта. Например,



2. В программном коде, чтобы реализовать ветвление применяется условный оператор IF THEN

Условный оператор позволяет выбирать и выполнять действия в зависимости от истинности некоторого условия. Имеется два варианта синтаксиса: В первом случае он имеет вид:

IF условие Then [операторы 1] [Else операторы 2]

Во втором случае оператор расположен на нескольких строках:

IF условие Then

[операторы]

[ElseIf условие – n Then

[операторы-n]...

[Else

[ИначеОператоры]]

End If

Здесь условие обязательно в обоих вариантах. Оно может быть числовым или строковым выражением со значениями TRUE или FALSE. Операторы 1 и операторы 2 это последовательности из одного или нескольких разделенных двоеточием операторов. По крайней мере одна из этих последовательностей должна быть непустой. Если условие истинно (TRUE), выполняется последовательность «операторы 1», если ложно, “операторы 2”.

Пример записи оператора:

‘Условный оператор в виде одной строки:

IF A>10 Then A=A+1: B=B+A: C=C+B ELSE C=A*B: A=C+2

‘тот же условный оператор в виде блока:

```

IF A>10 Then
  A=A+1: B=B+A
  C=C+B
ELSE C=A*B: A=C+2

```

End If

Примеры использования оператора условия

Пример.

Ввести X, вычислить F по формуле:

$$F = \begin{cases} X/2, & \text{если } X > 0 \\ (X+1)/2, & \text{если } X < 0 \end{cases}$$

Рис. Интерфейс приложения

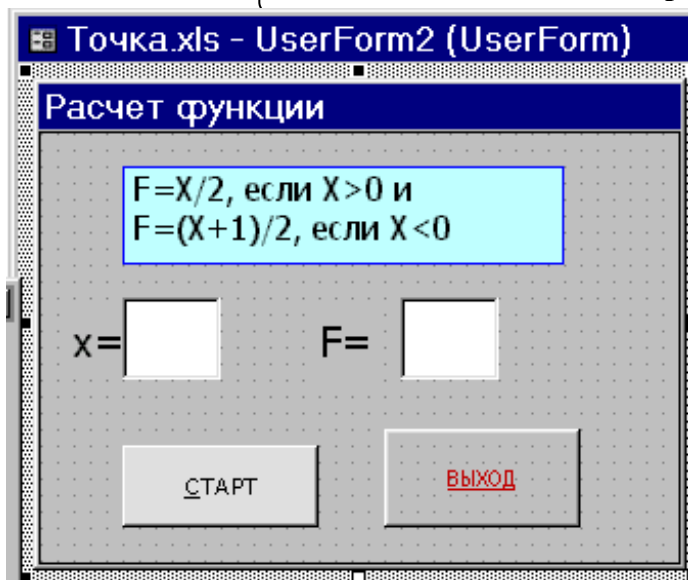
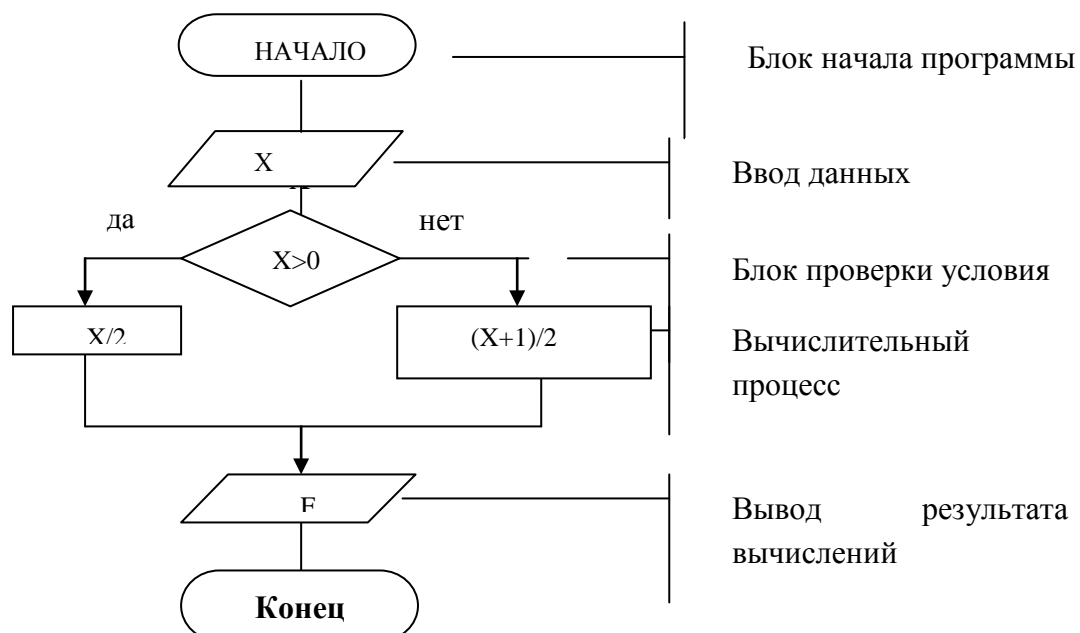


СХЕМА АЛГОРИТМА ПРОГРАММЫ



```

Private Sub CommandButton1_Click()
Dim x, F As Single
x = TextBox1.Value
    If x = 0 Then MsgBox "Функция не определена", vbCritical, "Расчет функции"
    If x > 0 Then F = x / 2 Else F = (x + 1) / 2
    TextBox2.Value = F
End Sub

Private Sub CommandButton2_Click()
Unload Me
End Sub

```

2 часть. Программирование с использованием объектов EXCEL.

Разрабатывать приложения в редакторе VBA можно и с использованием объектов приложений Office(объектов Excel, Word и др.) Рассмотрим разработку приложения примера2 с использованием объектов Excel. Будем использовать объекты:

- Worksheets() –для обозначения листа Excel
- Range() – для обозначения диапазона ячеек или одной ячейки
- свойство .Value для обращения к значению ячейки.

Т.е., если мы хотим записать в ячейку значение F программным способом, нужно написать в программе так:

Worksheets().Range().Value = F, где в скобках указать имя или номер листа EXCEL и адрес ячейки. Например, чтобы :

Присвоить ячейке C1 на листе “лист1” значение переменной F	<i>Нужно написать</i> Worksheets(“лист1”).Range(“C1”).Value = F
Изменить значение в ячейке A1 на листе “лист1”	WorkSheets(“лист1”).Range(“A1”).Value=3
Установить формулу для ячейки B1 на активном листе	Range(“B1”).Formula = “= - 5 + A1”

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ:

4. В Ячейку A1 ввести текст « Исходные данные»
5. В ячейку A2 ввести текст «X=»
6. В ячейку B2 ввести значение X.
7. В ячейку C1 ввести текст «результат при x>0»
8. В ячейку D1 ввести текст « результат при x<0»
9. Выполнить команду Сервис /Макрос/Редактор VBA
10. Выполнить команду Вставка/Модуль
11. Ввести имя модуля SUB LL()
12. Набрать текст программы:

```

Sub LL ()
X =Worksheets(1).Range(“B2”).Value
If X >0 Then

```

```

F= X /2
Worksheets(1).Range("C2").Value = f
Else
f=( X +1)/2: Worksheets(1).Range("D2").Value = f
End If
End Sub

```

ЗАДАНИЯ К ВЫПОЛНЕНИЮ:

Номер варианта	Выражение	Исходные данные
1	2	3
1	<p>Вычислить значение функции</p> $y = \begin{cases} 0, & \text{если } x = 0 \\ x^2, & \text{если } x \neq 0 \end{cases}$	x
2	<p>Вычислить значение функции</p> $y = \begin{cases} \sin x, & \text{если } x > 0 \\ 0, & \text{если } x = 0 \\ \cos x, & \text{если } x < 0 \end{cases}$	x
3	Ввести три целых числа и определить, сумма каких двух является наибольшей. Числа считать с рабочего листа и использовать мастер функций EXCEL.	A, b, c
4	Ввести три целых числа и определить, сумма каких двух является наибольшей. Числа считать с рабочего листа и использовать мастер функций EXCEL.	X, y, z
5	Ввести число X и определить, является ли оно четным.	
6	Ввести число N и определить делится ли оно без остатка на число M.	
7	$a = \begin{cases} (x+y)^2 - \sqrt{x \cdot y}, & \text{если } x \cdot y > 0 \\ (x+y)^2 + \sqrt{x \cdot y}, & \text{если } x \cdot y < 0 \\ (x+y)^2 + 1, & \text{если } x \cdot y = 0 \end{cases}$	
8	$b = \begin{cases} \ln(x/y) + (x^2 + y)^3, & \text{если } x \cdot y > 0 \\ \ln x/y + (x^2 + y)^3, & \text{если } x \cdot y < 0 \\ x + y, & \text{если } x = 0 \\ 0, & \text{если } y = 0 \end{cases}$	x, y
9	$c = \begin{cases} x^2 + y^2 + \sin x, & \text{если } x - y = 0 \\ (x - y)^2 + \cos x, & \text{если } x - y > 0 \\ (y - x)^2 + \operatorname{tg} x, & \text{если } x - y < 0 \end{cases}$	x, y

10	$d = \begin{cases} (x - y)^3 + \arctg x, & \text{åñëë } x > y \\ (y - x)^3 + \arctg x, & \text{åñëë } x < y \\ (x + y)^3 + 0.5, & \text{åñëë } x = y \end{cases}$	x, y
11	$e = \begin{cases} i \cdot \sqrt{a}, & \text{åñëë } i - \text{íå÷åðíå } , a > 0 \\ 0.5 \cdot i \cdot \sqrt{ a }, & \text{åñëë } i - \text{÷åðíå } , a < 0 \\ \sqrt{ i \cdot a }, & \text{èíå÷å} \end{cases}$	a, i

Лабораторная работа №4

VBA. ЦИКЛ С ПАРАМЕТРОМ (For...Next)

КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ.

1. Цикл For.

Оператор цикла For позволяет повторять группу операторов заданное число раз.
Синтаксис:

For *счётчик_цикла* =*начало* **TO** *конец* [**step** шаг]

Тело цикла

Next [*счётчик_цикла*]

Здесь *счётчик_цикла*—это числовая переменная. В начале выполнения цикла она принимает значение, задаваемое числовым выражением *начало*. Числовое выражение *конец*— задает заключительное выражение счётчика цикла. Числовое выражение шаг не обязательно и по умолчанию=1. *Тело цикла*— это последовательность операторов которая будет выполнена заданное число раз. Если шаг положителен, цикл завершится, когда впервые выполнится условие:

счётчик_цикла>*конец*

Если шаг цикла отрицателен, условие его завершения:

счётчик_цикла<*конец*

Это условие проверяется перед началом выполнения цикла, а затем—после каждого прибавления шага к счётчику цикла в операторе Next. Если оно выполнено, управление передается на оператор, следующий за Next, нет—выполняются операторы из тела цикла. Завершить цикл For...Next можно и с помощью оператора Exit For. Такие операторы могут быть расположены в тех местах тела цикла, где требуется из него выйти не дожидаясь выполнения условия завершения.

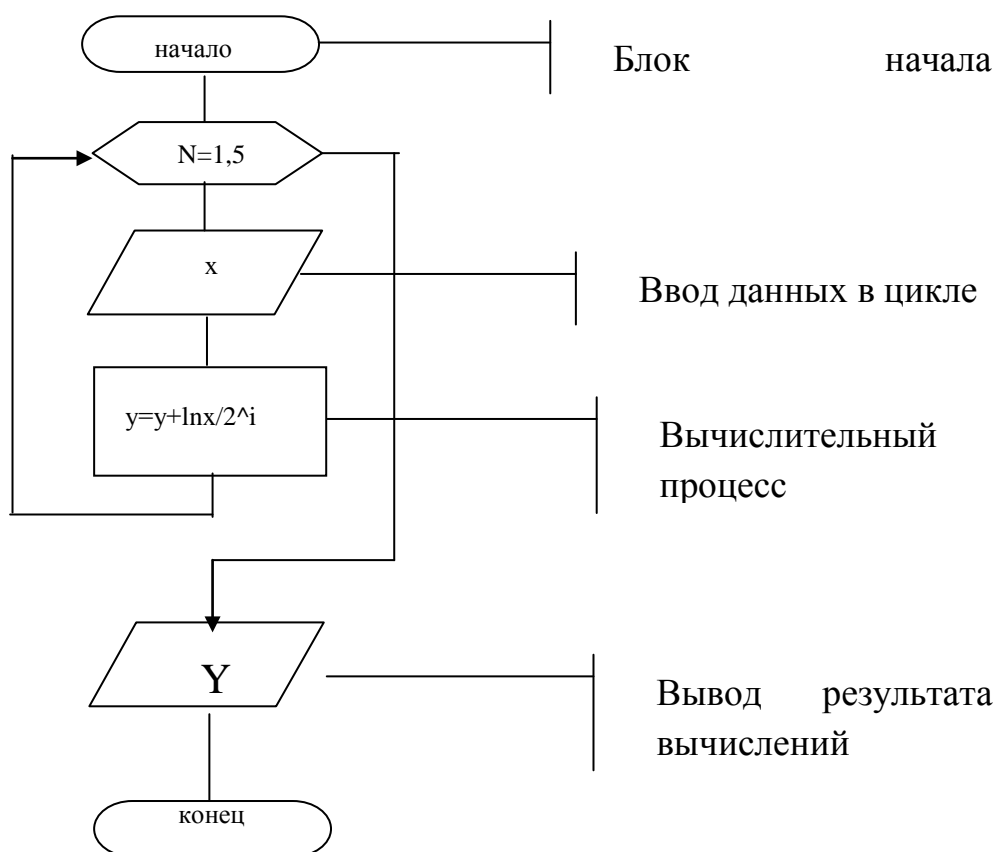
ПРИМЕР ИСПОЛЬЗОВАНИЯ ОПЕРАТОРА:

УСЛОВИЕ:

Вычислить значение функции:

$$Y = \sum_{i=1}^n \ln \frac{x}{2^n}, \text{ где } n=1,2,\dots,5$$

Схема алгоритма программы



ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ:

1. Заполнить диапазон ячеек A1:A5 значениями X.
2. Текст программы:

```

Sub mm ()
    N=5 : Y=0
    For i=1 to n
        X=Worksheets(1).Cells(i,1)
        Y=Y+log(x)/2^i
    Next i
    Worksheets(1).Range ("A6").Value = "результат"
    Worksheets(1).Range ("A7").Value = Y
End Sub

```

2.Массивы.

а) Описание массивов

Dim <имя массива>(<начальное значение индекса> **To** <конечное значение индекса>) **As** <тип элементов массива>

Пр: Dim A(1 To 10) As Integer - массив состоит из 10 элементов, тип каждого элемента -

Integer.

б) Обращение к элементу массива

Обращение к элементу массива осуществляется следующим образом: указывается имя массива, а затем в круглых скобках указывается номер элемента в массиве.

Пр: A(1)=5 - первому элементу массива A присваивается значение 5

A(17)=A(1) - 17-ому элементу массива A присваивается значение первого элемента массива A.

в) Ввод массивов

Массивы можно вводить как с листа Excel, так и используя встроенную функцию InputBox.

Пр: вводятся массивы A,B,C из 15 элементов:

For i=1 To 15

' Ввод массива A с листа Excel, используя свойство Range (элементы вводятся из столбца A, строки изменяются с 1 по 15.

A(i)=Worksheets(1).Range("A" & i).Value

' Ввод массива B с листа Excel, используя свойство Cells (строки изменяются с 1 по 15, столбец 2 (столбец B)

B(i)=Worksheets(1).Cells(i,2)

' Ввод массива C через функцию InputBox

C(i)=InputBox("Введите " & i & "ый элемент массива")

Next i

г) Решение задачи из примера через массивы

1. Заполнить диапазон ячеек A1:A5 значениями элементов массива.

2. Текст программы:

Sub mm ()

Dim A(1 to 5) As Integer

N=5

Y=0

For i=1 to n

A(i)=Worksheets(1).Cells(i,1)

Next i

For i=1 To n

Y=Y+log(A(i))/2^i

Next i

Worksheets(1).Range ("A6").Value = "результат"

Worksheets(1).Range ("A7").Value = Y

End Sub

ЗАДАНИЯ ДЛЯ ВЫПОЛНЕНИЯ

1. Найти количество положительных
2. Найти количество отрицательных чисел.
3. Найти сумму положительных
4. Найти сумму отрицательных чисел.
5. Найти количество элементов массива больших или равных заданному значению.
6. Найти произведение положительных элементов массива.
7. Найти сумму элементов, стоящих на четных местах .
8. Найти сумму элементов стоящих на нечетных местах.
9. Заменить отрицательные элементы нулями.
10. Обнулить элементы, имеющие четный индекс.
11. Найти произведение, элементов стоящих на четных местах.
12. Сформировать новый массив по правилу: $c[i]=x*\sqrt{a[i]}$, где x — некоторая постоянная.

3. Двумерные массивы

а) Описание массивов

Dim <имя массива>(<начальное значение индекса по строкам> **To** <конечное значение индекса по строкам> ,
< начальное значение индекса по столбцам> **To** < конечное значение индекса по столбцам>)
As <тип элементов массива>

Пр: Dim A(1 To 10, 1 to 5) As Integer - массив состоит из 50 элементов (10 строк и 5 столбцов), тип каждого элемента - Integer.

б) Обращение к элементу массива

Обращение к элементу массива осуществляется следующим образом: указывается имя массива, а затем в круглых скобках через запятую указывается номер строки и номер столбца, где размещен элемент в массиве.

Пр: A(1,4)=15 - элементу, находящемуся в первой строке и четвертом столбце массива A присваивается значение 15

в) Ввод массивов

Массивы можно вводить как с листа Excel, так и используя встроенную функцию InputBox.

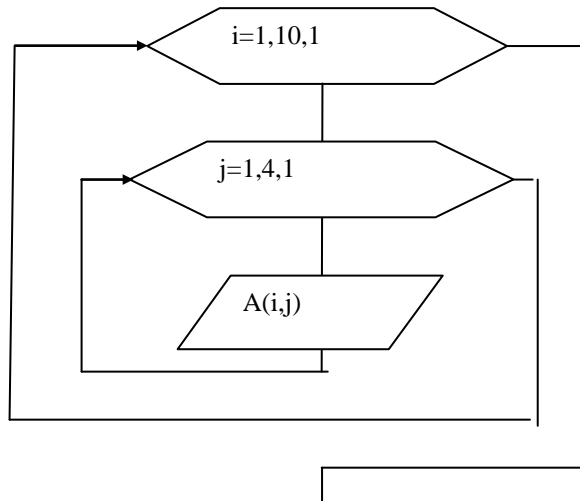
- 1) for i=1 to 10
for j=1 to 4
A(i,j)=InputBox("Введите A(" & i & "," & j & ")")
next j
next i
- 2) for i=1 to 10

```

for j=1 to 4
  A(i,j)=Worksheets(1).Cells(i,j)
next j
next i

```

г) Изображение ввода двумерных массивов в блок-схемах



СОДЕРЖАНИЕ ОТЧЕТА

1. Название лабораторной работы.
2. Цель работы.
3. Задание
4. Схема алгоритма.
5. Текст программы.
6. Выводы.

ЗАДАНИЯ ДЛЯ ВЫПОЛНЕНИЯ

- 1) Создать одномерный массив из 20 целочисленных значений. Найти наибольшее среди положительных элементов, стоящих на четных местах, и вывести его значение и индекс данного элемента.
- 2) Создать одномерный целочисленный массив из 20 значений. Подсчитать среднее арифметическое значение всех элементов и вывести на экран все значения, меньшие среднего арифметического.
- 3) Создать одномерный целочисленный массив из 20 значений. Расставить в нем элементы в обратном порядке.
- 4) Создать одномерный целочисленный массив из 20 значений. Поменять местами каждую пару чисел, например, A1 с A2, A3 с A4 и т.д.
- 5) Создать одномерный целочисленный массив из 20 значений. Найти разность между каждыми двумя последовательными значениями, т.е. между B1 и B2, B2 и B3, и т.д.
- 6) Создать одномерный массив из 20 целочисленных значений. Найти минимальную сумму каждой пары значений, т.е. C1 и C2, C3 и C4.
- 7) Создать два одномерных массива по 10 целочисленных значений каждый. Сформировать третий массив из 20 элементов следующим образом: на нечетные места ставить элементы из первого массива, на четные места – из второго.
- 8) Создавать одномерный массив из 20 целочисленных значений. Элементы, стоящие на нечетных местах, расставить в обратном порядке.

- 9) Создать одномерный массив С из 10 целочисленных значений. Сформировать новый массив С1, где каждый элемент будет вычисляться по формуле $C1[I] = X * \sqrt{C[I]}$, где X – некоторая постоянная.
- 10) Создать одномерный целочисленный массив из 20 значений. Найти максимальное значение модуля разности между каждой парой элементов, т.е. между D1 и D2, D3 и D4.
- 11) Создать три одномерных массива по 10 целочисленных значений. Сравнить по 3 I – тых элемента и вывести максимальное значение.
- 12) Создать два одномерных массива по 20 целочисленных значений. Совершить обмен данными: в одном массиве с 1-ого по 10- ый элемент, в другом с 11 – го по 20 – ый элемент, т.е. X1 на Y11, Y11 на X1 и т.д.
- 13) Создать одномерный целочисленный массив из 20 значений. Найти сумму элементов, стоящих на четных местах.
- 14) Создать одномерный массив из 20 целочисленных значений. Найти произведение положительных элементов массива.
- 15) Создать одномерный массив из 20 целочисленных значений. Расставить в нем в том же порядке сначала положительные элементы, затем отрицательные, затем нулевые.
- 16) Создать одномерный массив из 20 целочисленных значений. Все четные элементы расставить по возрастанию.
- 17) Создать одномерный массив из 20 значений. Расставить все элементы по убыванию.
- 18) Создать одномерный массив из 20 целочисленных значений. Найти минимальное среди них, вывести его и индекс данного элемента на экран.
- 19) Создать одномерный массив из 20 целочисленных значений. Найти минимальное и максимальное значения, вывести их разность на экран.
- 20) Создать одномерный массив из 20 целочисленных значений. Найти наибольшее среди отрицательных и вывести его индекс на экран.
21. Дана целочисленная прямоугольная матрица. Определить количество строк, содержащих хотя бы один нулевой элемент.
22. Дан массив размерностью n x m. Заменить все положительные элементы на 1, все отрицательные на -1.
23. Дан массив размерностью n x m. Подсчитать сумму элементов, у которых сумма номера строки и номера столбца равна n+1.
24. Дан массив размерностью n x m. Подсчитать сумму элементов, у которых модуль разности номера строки столбца равен 1.
25. Дан массив размерностью n x m. Подсчитать сумму элементов, которые меньше номера своей строки.
26. Дан массив размерностью n x m. Подсчитать сумму элементов, которые больше номера своего столбца и строки.
27. Дан массив размерностью n x m. Подсчитать сумму положительных элементов и распечатать их номера.
28. Дан массив размерностью n x m. Подсчитать сумму нечетных элементов и распечатать их номера.
29. Дан массив размерностью n x m. Подсчитать сумму элементов, имеющих одинаковые остатки при делении на 7 и на 2, и распечатать номера таких элементов.
30. Дан массив размерностью n x m. Преобразовать элементы массива по следующему правилу: если элемент четный то разделить его на 2, если нечетный – заменить его остатком от деления на 3.
31. . Дан массив размерностью n x m. Преобразовать элементы массива по следующему правилу: если элемент положительный, то умножить его на 2, а если отрицательный – поменять знак на противоположный.
32. . Дан массив размерностью n x m. Преобразовать элементы массива по следующему правилу: если элемент четный, то прибавить к нему 1, если нечетный – умножить на 2.

Лабораторная работа № 5

ЦЕЛЬ РАБОТЫ: Изучение и применение операторов цикла с предусловием и постусловием.

1. Операторы цикла с предусловием:

а) Do While <условие>
<операторы>
[Exit Do]
< операторы >

б) While <условие>
< операторы >
Wend

Loop

Работа: **Тело цикла** выполняется в том случае, если **условие** имеет значение **истина**. Если **условие ложно**, то управление в программе передается оператору, следующему за оператором цикла.

На блок схеме данные операторы изображаются следующим образом

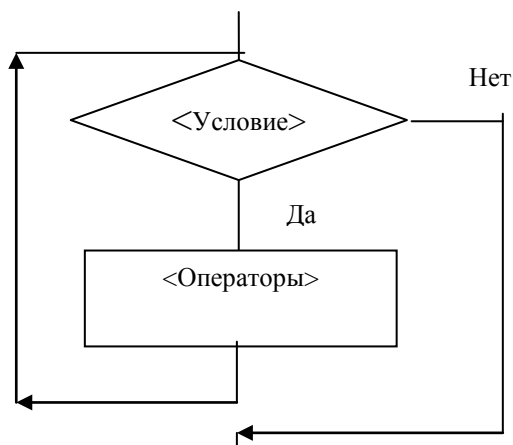


Рисунок 1 – Операторы с предусловием **Do While <условие><тело цикла> Loop; While <условие><тело цикла> Loop**

Кроме описанных выше операторов с предусловием существует еще один оператор с предусловием:

Do Until <условие>
<операторы>
[Exit Do]
< операторы >
Loop

Работа: **Тело цикла** выполняется в том случае, если **условие** имеет значение **ложь**. Если **условие истинно**, то управление в программе передается оператору, следующему за оператором цикла.

На блок схеме этот оператор отображается следующим образом

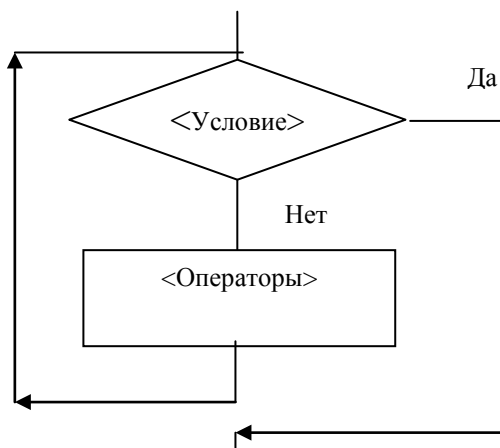


Рисунок 2 – Операторы с предусловие **Do Until<условие>< тело цикла> Loop**

Пример: *Вводить числа, пока их сумма не превысит введенного числа m.*

Данная задача решается с применением оператора с предусловием **Do While ...Loop**.

```

Public Sub uuu()
  Dim x As Integer
  Dim m As Integer
  Dim s As Integer
  Dim i As Integer
  m = InputBox("Введите число m")
  MsgBox ("Вводите числа")
  i = 1
  s = InputBox("Введите 1 число")
  Do While s <= m
    i = i + 1
    x = InputBox("Введите " & i & "число")
    s = s + x
  Loop
  MsgBox ("Количество введенных чисел " & i)
End Sub
  
```

Далее приведен код программы той же задачи, но с использованием цикла **Do Until ... Loop**.

```

Public Sub uuu()
  Dim x As Integer
  Dim m As Integer
  Dim s As Integer
  Dim i As Integer
  m = InputBox("Введите число m")
  MsgBox ("Вводите числа")
  i = 1
  s = InputBox("Введите 1 число")
  Do Until s > m
    i = i + 1
    x = InputBox("Введите " & i & "число")
    s = s + x
  Loop
  MsgBox ("Количество введенных чисел " & i)
End Sub
  
```

2. Операторы цикла с постусловием

а) Do

<операторы>

[Exit Do]

< операторы >

Loop While <условие>

Работа: Выполняется **тело цикла**. Затем вычисляется **условие**. Если **условие** имеет значение **истина**, то опять выполняется **тело цикла**. Если **условие ложно**, то управление в программе передается оператору, следующему за оператором цикла.

На блок схеме данные оператор изображаются следующим образом

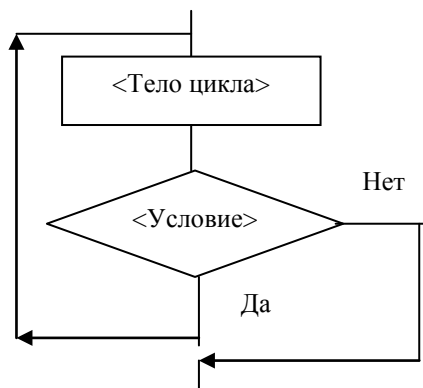


Рисунок 3 – Операторы с постусловием **Do** <тело цикла> **Loop While**<условие><

Далее приведен код программы, с использованием цикла **Do ...Loop While**.

```
Public Sub uuu()  
    Dim x As Integer  
    Dim m As Integer  
    Dim s As Integer  
    Dim i As Integer  
    m = InputBox("Введите число m")  
    MsgBox ("Вводите числа")  
    i = 0  
    s = 0  
    Do  
        i = i + 1  
        x = InputBox("Введите " & i & "число")  
        s = s + x  
    Loop While s <= m  
    MsgBox ("Количество введенных чисел " & i)  
End Sub
```

б) Do

<операторы>

[Exit Do]

< операторы >

Loop Until <условие>

Работа: Выполняется **тело цикла**. Затем вычисляется **условие**. Если **условие** имеет значение **ложь**, то опять выполняется **тело цикла**. Если **условие истинно**, то управление в программе передается оператору, следующему за оператором цикла.

На блок схеме данные оператор изображаются следующим образом

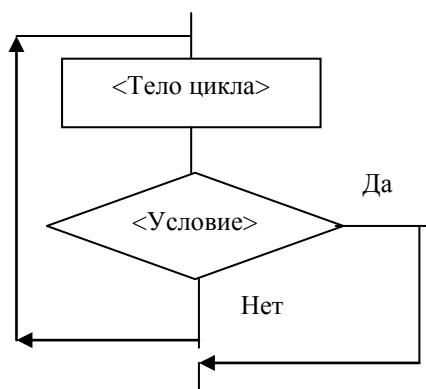


Рисунок 4 – Операторы с постусловие **Do <тело цикла> Loop Until<условие><**

Ниже находится код все той же задачи, но с использованием последнего описанного цикла.

```

Public Sub uuu()
  Dim x As Integer
  Dim m As Integer
  Dim s As Integer
  Dim i As Integer
  m = InputBox("Введите число m")
  MsgBox ("Вводите числа")
  i = 0
  s = 0
  Do
    i = i + 1
    x = InputBox("Введите " & i & "число")
    s = s + x
  Loop Until s > m
  MsgBox ("Количество введенных чисел " & i)
End Sub
  
```

ЗАДАНИЕ:

Решить задачу согласно своему варианту, используя все 5 способов описания операторов цикла с предусловием и постусловием

Вариант	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.
Номера	1,	2,	5,	6,	8,	9,	10,	11,	12,	13,	14,	7,	4,	3,

1. Начиная с какого номера n имеет место неравенство $n! > x^n$, x - любое действительное число.
2. Вводить последовательность до тех пор, пока не встретятся три подряд идущих положительных числа. Тогда прервать ввод и сообщить, сколько во введенной последовательности было всего чисел.
3. Вводить последовательность до тех пор, пока не встретятся три подряд идущих положительных числа. Тогда прервать ввод и сообщить, сколько во введенной последовательности было положительных чисел.
4. Вводить последовательность до тех пор, пока не встретятся три подряд идущих положительных числа. Тогда прервать ввод и сообщить, сколько во введенной последовательности было отрицательных чисел.

5. Определить сколько натуральных подряд идущих четных чисел нужно сложить (найти минимальное число таких слагаемых), чтобы их сумма была больше введенного числа.
6. Дано натуральное N и первый член бесконечного ряда: $Y_1=1$. Вычислить сумму членов бесконечного ряда, образованного по следующему рекуррентному соотношению: $Y_i=2*Y_{i-1}$ (то есть $S=1+2+4+8+16+\dots$). Вычисление суммы продолжать до тех пор, пока соблюдается условие $|Y_i - Y_{i-1}| < N$.
7. Последовательно вводятся числа до тех пор, пока во введенной совокупности не окажется три нуля. Вывести количество введенных чисел.
8. Дано натуральное число.
- Верно ли, что сумма его цифр меньше A ?
 - Верно ли, что произведение его цифр больше B ?
 - Верно ли, что это число k -значное?
9. Дано натуральное число. Определить номер цифры 3 в нем, считая от конца числа. Если такой цифры нет, ответом должно быть число 0, если таких цифр в числе несколько — должен быть определен номер самой правой из них.
10. Дано натуральное число. Если в нем есть цифры "2" и "5", то определить, какая из них расположена в числе левее. Если одна или обе эти цифры встречаются в числе несколько раз, то должны быть рассмотрены самые левые из одинаковых цифр.
11. Дано натуральное число. Определить:
- количество цифр в нем;
 - сумму его цифр;
 - произведение его цифр;
 - среднее арифметическое его цифр;
12. Дано натуральное число. Определить:
- сумму квадратов его цифр;
 - сумму кубов его цифр;
 - его первую цифру;
 - сумму его первой и последней цифр.
13. Дано натуральное число.
- Верно ли, что его первая цифра не превышает 6?
 - Верно ли, что оно начинается и заканчивается одной и той же цифрой?
 - Определить, какая из его цифр больше: первая или последняя.
14. Дано натуральное число. Определить:
- количество цифр "3" в нем;
 - сколько раз в нем встречается цифра, равная последней;
 - количество четных цифр в нем.

Лабораторная работа № 6

VBA. Переключатели.

Цель работы: изучить свойства элемента управления **Переключатель**; использовать его для решения задач.

Краткие теоретические сведения.

Элемент управления **OptionButton** (переключатель) создается с помощью кнопки **Переключатель** (OptionButton). Он позволяет выбрать, один из нескольких взаимоисключающих параметров или действий. Переключатели обычно отображаются группами, обеспечивая возможность выбора альтернативного варианта.

Приведем наиболее часто используемые свойства элемента управления **OptionButton**.

Value	Возвращает True, если переключатель выбран и False в противном случае
Enabled	Допустимые значения: True (пользователь может выбрать переключатель) и False (в противном случае)
Visible	Допустимые значения: True (переключатель отображается во время выполнения программы) и False (в противном случае)
Caption	Надпись, отображаемая рядом с переключателем

Основными событиями переключателя являются события **Click** и **Change**.

Пример:

Разработать программу выполнения одной из четырех арифметических операций над двумя числами по выбору пользователя. Исполняемая операция устанавливается за счет выбора соответствующего переключателя.

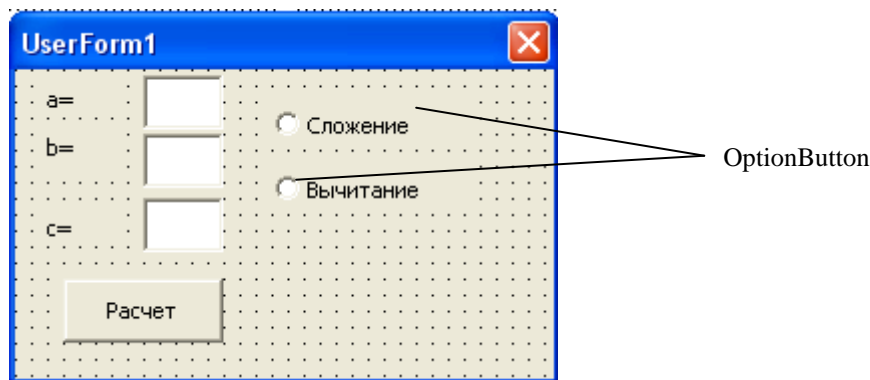


Рис.: Использование переключателей.

```
Private Sub CommandButton1_Click()  
Dim a As Integer, b As Integer, c As Integer  
a = TextBox1.Value  
b = TextBox2.Value  
If OptionButton1.Value = True Then  
    c = a + b  
End If
```

```
If OptionButton2.Value = True Then
    c = a - b
End If
TextBox3.Value = c
End Sub
```

Задания к лабораторной работе.

Создать пользовательскую форму и написать программу для решения следующей задачи с использованием *переключателей*:

1. Выбирается число от 1 до 4, определяющее пору года. Вывести название этой поры.
2. Выбирается число от 1 до 7, определяющее день недели. Дать название этого дня.
3. Выбирается число от 1 до 5. Дать название этого числа.
4. Вводится нецелое число. Вывести либо его целую часть, либо дробную в зависимости от выбора пользователя.
5. Банк предлагает три вида срочных вкладов: на 3 месяца под 27 %, на 6 месяцев под 29 % и на год под 30 %. Вкладчик положил N у. е. на один из срочных вкладов. Какую сумму он получит по истечении срока?
6. Задано расстояние в метрах. Пересчитать это расстояние в километрах, милях, футах или ярдах на выбор пользователя (1 миля=1,609 километра, 1 метр=1,094 ярда, 1 метр=3,281 фута).
7. Дан объем в литрах. Пересчитать этот объем в пинтах, галлонах, бушелях и квартах (английские меры объема жидких и сыпучих тел) на выбор пользователя (1 литр=1,706 пинты, 1 литр=0,220 галлона, 1 бушель=36,35 литра, 1 кварта=1,136 литра).
8. Дана масса в килограммах. Пересчитать эту массу в пудах, фунтах, центнерах или тоннах на выбор пользователя (1 пуд=16,38 кг, 1 фунт=0,409 кг, 1 т=1000 кг, 1 ц=100 кг.).
9. Дано расстояние в метрах. Пересчитать его в верстах, саженьях, аршинах или вершках на выбор пользователя (1 верста=1,067 км, 1 сажень=2,134 м, 1 аршин=0,7112 м, 1 вершок=4,445 см.).

Лабораторная работа №7.

VBA: списки.

Цель работы: изучить свойства, события и методы элемента управления Список; использовать списки при решении задач.

Краткие теоретические сведения.

Элемент управления ListBox (список) создается с помощью кнопки **Список** (ListBox). Элемент управления ListBox применяется для хранения списка значений. Из списка пользователь может выбрать одно или несколько значений, которые в последующем будут использоваться в тексте программы.

Наиболее часто используемые свойства элемента управления ListBox.

ListIndex	Возвращает номер текущего элемента списка. Нумерация элементов списка начинается с нуля
ListCount	Возвращает число элементов списка
TopIndex	Возвращает элемент списка с наибольшим номером
ColumnCount	Устанавливает число столбцов в списке
TextColumn	Устанавливает столбец в списке, элемент которого возвращается свойством Text
Enabled	Допустимые значения: True (запрещен выбор значения из списка пользователем) и False (в противном случае)
Text	Возвращает выбранный в списке элемент
List	Возвращает элемент списка, стоящий на пересечении указанных строки и столбца. Синтаксис: List(row, column)
RowSource	Устанавливает диапазон, содержащий элементы списка
ControlSource	Устанавливает диапазон (ячейку), куда возвращается выбранный элемент из списка
MultiSelect	Устанавливает способ выбора элементов списка. Допустимые значения: <ul style="list-style-type: none">- fmMultiSelectSingle (выбор только одного элемента)- fmMultiSelectMulti (разрешен выбор нескольких элементов посредством либо щелчка, либо нажатием клавиши <Пробел>)- fmMultiSelectExtended (разрешено использование клавиши <Shift> при выборе ряда последовательных элементов списка)
Selected	Допустимые значения: True (если элемент списка выбран) и False (в противном случае). Используется для определения выделенного текста, когда свойство MultiSelect имеет значение fmMultiSelectMulti или fmMultiSelectExtended

Наиболее часто используемые методы элемента управления ListBox.

Clear	Удаляет все элементы из списка
RemoveItem	Удаляет из списка элементы с указанным номером. Синтаксис: Remove Item (index) index — номер, удаляемого из списка элемента

AddItem	Добавляет элемент в список. Синтаксис: AddItem ([item [, arIndex]]) -item — элемент, добавляемый в список -varIndex — номер добавляемого элемента
---------	--

Заполнить список можно одним из следующих способов:

Поэлементно, если список состоит из одной колонки	With ListBox1 .AddItem "Июнь" .AddItem "Июль" .AddItem "Август" End With
Массивом, если список состоит из одной колонки	With ListBox1 .List = Array("Июнь", "Июль", Август") .ListIndex = 1 End With
Из диапазона A1 : B4, в который предварительно введены элементы списка. Результат выбора (индекс выбранной строки) выводится в ячейку C1.	With ListBox1 .ColumnCount = 2 .RowSource = "A1:B4" .ControlSource = "C1" End With
Поэлементно, если список состоит из нескольких колонок, например двух	With ListBox1 .ColumnCount = 2 .AddItem "Июнь" .List(0, 1) = "Сессия" .AddItem "Июль" .List(1, 1) = "Каникулы" .AddItem "Август" .List (2, 1) = "Каникулы" End With
Массивом, если список состоит из нескольких колонок, например двух	Dim A (2, 1) As String A(0, 0) = "Июнь" A(0, 1) = "Сессия" A(1, 0) = "Июль" A(1, 1) = "Каникулы" A(2, 0) = "Август" A(2, 1) = "Каникулы" With ListBox1 .ColumnCount = 2 .List = A End With

Задания к лабораторной работе.

Задание 1. Разработать программу, содержащую одностолбцовый список.

1. Дан линейный массив. Отсортировать его методом пузырька. Вывести в один список – исходный массив, в другой- отсортированный.
2. Дан линейный массив. Заменить четные числа на 1, нечетные – на -1. Вывести в один список – исходный массив, в другой - преобразованный.
3. Дан линейный массив. Вывести в один список – исходный массив, в другой – только элементы, кратные трем.
4. Вычислить $\frac{x^2}{2}, \frac{x^3}{3}, \dots, \frac{x^{11}}{11}$ для указанного значения x .
5. Вывести члены арифметической прогрессии. Значение первого члена, разность и количество членов задаются (формула n -го члена: $a_n = a_1 + d(n-1)$).
6. Вывести члены геометрической прогрессии. Значение первого члена, знаменатель и количество членов задаются (формула n -го члена: $b_n = b_1 q^{n-1}$).

Задание 2. Разработать программу, содержащую многостолбцовый список.

1. Рассчитать таблицу значений функции $y = \sqrt{x^2 + k^2}$, где x меняется от -2 до 2 с шагом 0.1, а k – параметр, задаваемый пользователем. Таблицу поместить в двухстолбцовый список.
2. Сумма в Р у.е. положена в банк. Ежегодный прирост составляет x % годовых. Вывести стоимость капитала в конце каждого года. Первоначальная сумма, процент прироста и срок задаются.
3. Составить таблицу перевода километров в мили на интервале от 10 до 50 с шагом 5 (1 миля=1,609 километра).
4. Составить таблицу перевода метров в ярды на интервале от 2 до 10 с шагом 0.5 (1 метр=1,094 ярда).
5. Составить таблицу перевода метров в футы на интервале от 10 до 50 с шагом 5 (1 метр=3,281 фута).
6. Составить таблицу квадратных корней из чисел от a до b с шагом 0.1. Значения a и b задаются ($a < b$).

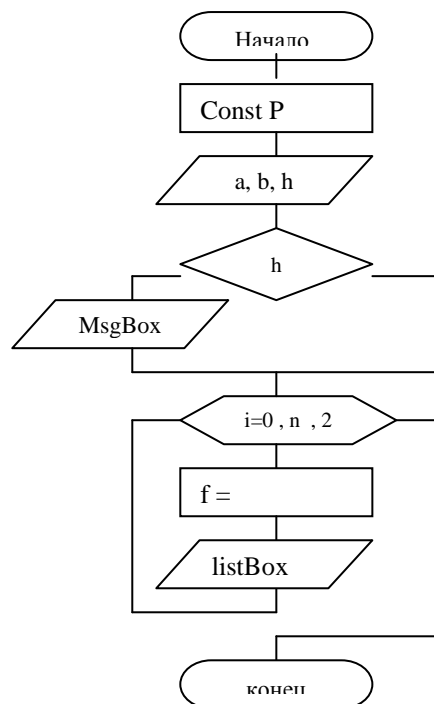
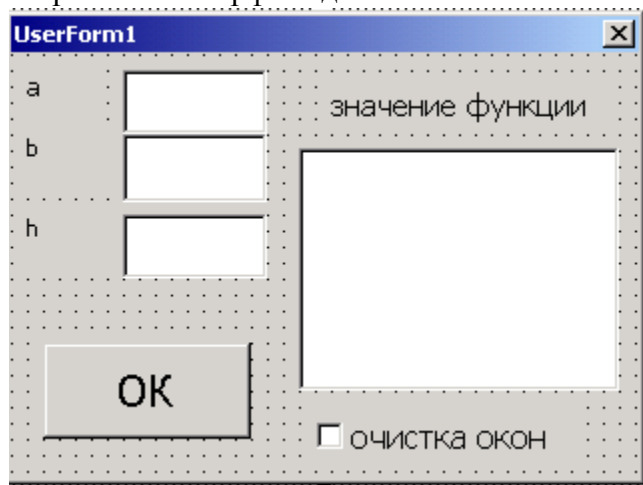
Задание 3. Разработать программу, содержащую несколько списков, осуществляющую выбор из списка.

1. Разработать программу, «расспрашивающее» покупателя авиабилета о характере заказа: пункт назначения, расположение кресла (в середине, у прохода, у окна), тип салона (для курящих или не курящих). Рассчитайте стоимость билета с учетом надбавок за расположение кресла и типа салона.
2. Разработать программу, помогающую посетителю кафе выбрать из списка понравившиеся ему блюда. Организуйте два списка: предлагаемые блюда, выбранные блюда. Выбор строки в любом списке приводит к ее перемещению в соседний список. Рассчитайте стоимость выбранных блюд.
3. На трех заводах «Альфа», «Плутон» и «Рубин» иногда происходят аварии. При выборе завода в первом списке, во втором вывести сведения о количестве аварий за каждый год (взять последние 4 года).
4. В двух списках записаны числа, указывающие первый член и разность арифметической прогрессии. Получить в третьем списке 10 первых членов арифметической прогрессии.
5. В двух списках записаны числа, указывающие первый член и знаменатель геометрической прогрессии. Получить в третьем списке 10 первых членов геометрической прогрессии.

Пример заполнения списка на примере расчета значений функции $f = a * \sin(x) + b * \cos(x)$ при x , принадлежащему отрезку от 0 до π с шагом h .

Блок-схема решения задачи:

Разработан интерфейс диалогового окна:



```

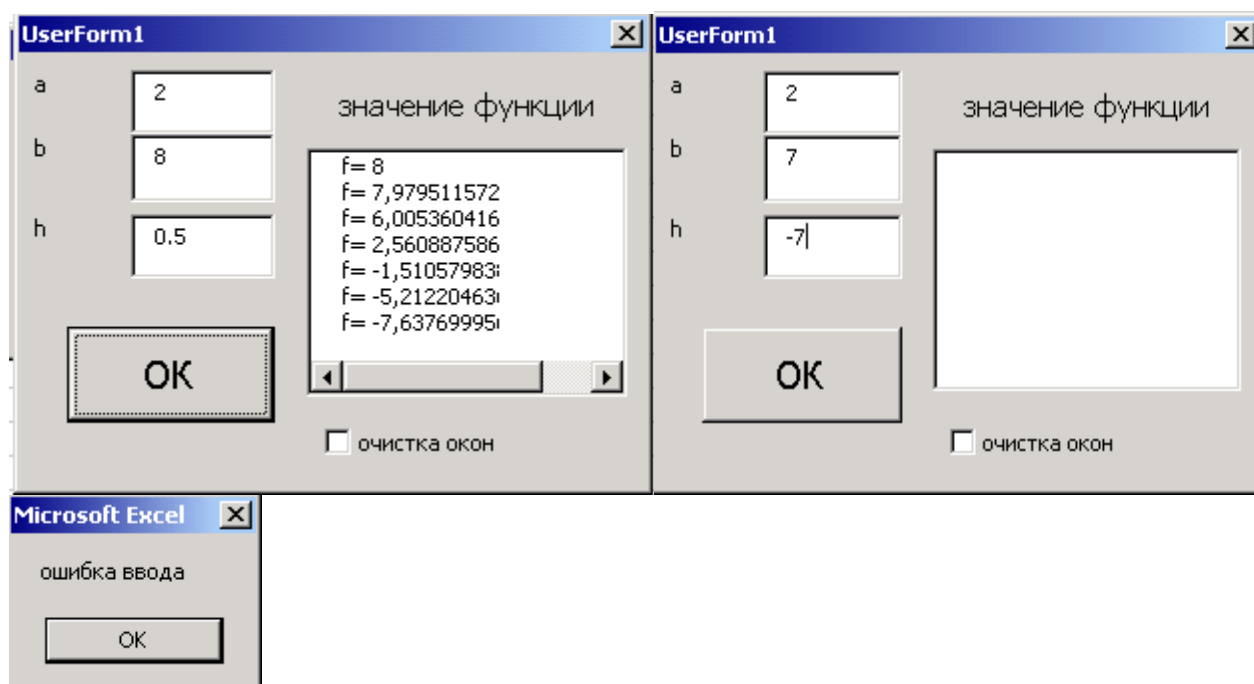
Private Sub CommandButton1_Click() ' процедура расчета
Dim a As Double, b As Double, h As Double
Dim f As Double
Const p = 3.14159
a = Val(TextBox1.Text) ' присвоение числового значения переменной a
b = Val(TextBox2.Text) ' присвоение числового значения переменной b
h = Val(TextBox3.Text) ' присвоение числового значения переменной h
If h < 0 Or h > p Then ' проверка ввода переменной h
MsgBox "ошибка ввода"
TextBox3.Text = " "
TextBox3.SetFocus
End If
For x = 0 To p Step h ' расчет значений функции
f = a * Sin(x) + b * Cos(x)
ListBox1.ColumnCount = 2 ' определение количества столбцов в ListBox1
ListBox1.AddItem " f= " & Cdbl(f) ' вывод вычислений
Next
End Sub
  
```

```

Private Sub CheckBox1_Click() ' процедура очистки окон
TextBox1.Text = "" ' очистка окна TextBox1
TextBox2.Text = "" ' очистка окна TextBox2
TextBox3.Text = "" ' очистка окна TextBox3
ListBox1.Clear ' очистка окна ListBox1
TextBox1.SetFocus ' перемещает фокус (курсор) в окно TextBox1
End Sub
  
```


Ввод данных осуществляется в два окна TextBox1, TextBox2 TextBox3. При нажатии на командную кнопку CommandButton1 (“OK”), происходит вывод значений функции в окно ListBox1. Пользователь может повторять расчет с новыми значениями, для этого необходимо очистить все окна ввода и выводы путем нажатия на окно CheckBox1.

Результаты тестирования:



ЛАБОРАТОРНАЯ РАБОТА № 8

Подпрограммы и их применение

Теоретические сведения

Подпрограмма — программа, реализующая вспомогательный алгоритм. *Основная программа* — программа, реализующая основной алгоритм решения задачи и содержащая в себе обращения к подпрограммам. В VBA существуют два типа подпрограмм:

- подпрограммы-функции
- подпрограммы-процедуры.

Отличие функции от процедуры заключается в том, что результатом исполнения операторов, образующих тело функции, всегда является некоторое единственное значение, поэтому обращение к функции можно использовать в соответствующих выражениях наряду с переменными и константами.

Таблица 15

Описание	Подпрограмма-процедура	Подпрограмма-функция
	Sub <имя процедуры> [(<i><список параметров></i>)] <операторы> [Exit Sub] <операторы> End Sub	Function <имя функции> [(<i><список параметров></i>)] [<i>As</i> <тип функции>] <операторы> [Exit Function] <операторы> <имя функции> = <выражение> End Function
Вызов основной программы	в 1) <имя процедуры> <список аргументов>; 2) Call <имя процедуры> [(<i><список аргументов></i>)]	<имя функции> (<список аргументов>)

Sub, End Sub – служебные слова VBA;

<имя процедуры> – имя процедуры, удовлетворяющее стандартным правилам именования;

<список аргументов> – список переменных, представляющих аргументы, которые передаются в процедуру Sub при её вызове. Имена переменных разделяются запятой;

<Инструкции> – любой набор команд VBA;

Exit Sub – инструкция, выполнение которой приводит к выходу из процедуры.

Синтаксис элемента <список аргументов>:

[ByVal | ByRef] <Имя переменной> [*As* <Тип>]

ByVal – ключевое слово, указывающее, что аргумент передается по значению;

ByRef – ключевое слово, указывающее, что аргумент передается по ссылке.

Описание **ByRef** используется в VBA по умолчанию;

<Имя переменной> – имя переменной, удовлетворяющее стандартным правилам именования переменных;

<Тип> – тип данных аргумента, переданного в процедуру;

В качестве результата процедура может возвращать в вызывающую программу множество простых или структурированных величин или не возвращать никаких значений. Среди параметров процедуры указываются как аргументы, так и результаты.

Обращение к процедуре — отдельный оператор.

Обращение к функции является операндом в выражении. Подпрограмма-функция вызывается в выражении по своему имени, за которым следует вписок аргументов в скобках.

При обращении к подпрограмме происходит *передача* ей *аргументов по ссылке* (если формальный параметр является параметром-переменной, описан как ByRef) или *по значению* (является параметром-значением, описан как ByVal).

Для того, чтобы понять, в каких случаях использовать тот или иной тип передачи аргументов, рассмотрим, как осуществляется замена формальных параметров на фактические в момент обращения к подпрограмме.

Если параметр определен как *параметр-значение* (с помощью ключевого слова ByVal), то перед вызовом подпрограммы это значение вычисляется, полученный результат копируется во временную память и передается подпрограмме. Важно учесть, что даже если в качестве фактического параметра указано простейшее выражение в виде переменной или константы, все равно подпрограмме будет передана лишь копия переменной (константы). Любые возможные изменения в подпрограмме параметра-значения никак не воспринимаются вызывающей подпрограммой, так как в этом случае изменяется копия фактического параметра.

Если параметр определен как *параметр-переменная* (по умолчанию или с помощью ключевого слова ByRef), то при вызове подпрограммы передается сама переменная, а не ее копия. Изменение параметра-переменной приводит к изменению самого фактического параметра в вызывающей подпрограмме.

Если в качестве фактического параметра используется константа, транслятор блокирует любые присваивания константе нового значения в теле подпрограммы.

Пример. Построить график функции $y = \frac{\sin x}{x^2 + 1}$ на отрезке [a;b] с шагом h. Построение графика организовать процедурой.

<pre>Public Sub prog7() Dim a As Double, b As Double Dim h As Double Worksheets(1).Range("A:B").Select Selection.Clear Worksheets(1).ChartObjects.Delete a = CDBl (InputBox("Введите a")) b = CDBl (InputBox("Введите b")) h = CDBl (InputBox("Введите h")) j = 1 For i = a To b Step h Worksheets(1).Range("A" & j) = i Worksheets(1).Range("B" & j) = Sin(i) / (i ^ 2 + 1) j = j + 1 Next i график End Sub</pre>	<p>Выделение двух столбцов (A и B) первого листа</p> <p>Очистка от данных выделенного диапазона</p> <p>Удаление с листа имеющихся диаграмм</p> <p>j используется для задания номера строки при выводе данных</p> <p>В цикле For...Next выводятся на лист Excel данные для построения диаграммы (в столбец A – значения аргумента, в столбец B – значения функции для соответствующего аргумента)</p> <p>Вызывается процедура график</p>
<pre>Sub график() n=Application.CountA(Worksheets (1).Range("A:A")) Range("A1:B" & CStr(n)).Select Charts.Add ActiveChart.ChartType xlXYScatterSmoothNoMarkers</pre>	<p>При создании процедуры график() вначале был создан макрос в Excel, а затем полученный макрос отредактирован. В частности, первая команда в данной процедуре определяет количество заполненных строк на листе в столбце A и это значение присваивается переменной n.</p>

```

ActiveChart.SetSourceData
Source:=Sheets("Лист1").Range("A1:B
" & CStr(n)), PlotBy:= xlColumns
ActiveChart.Location
Where:=xlLocationAsObject,
Name:="Лист1"
End Sub

```

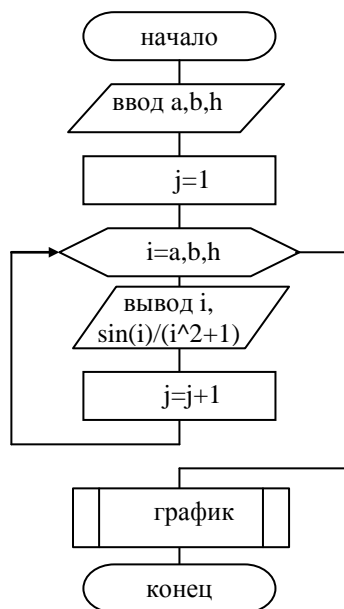


Рисунок 13 - Блок-схема программы prog7

Блок-схема макроса не изображается.

Пример. Вычислить сумму членов ряда $\sum_{i=1}^n \frac{1}{i!}$, где $i!$ – факториал числа i (произведение натуральных чисел от 1 до i).

```

Public Sub prog6()
Dim i As Integer, n As Integer
Dim s As Double
n = CInt(InputBox("Введите n"))
For i = 1 To n
s = s + 1 / faktor(i)
Next
MsgBox s
End Sub

```

```

Public Function faktor(x As
Integer) As Long
faktor = 1
For i = 1 To x
faktor = faktor * i
Next i
End Function

```

При вычислении искомой суммы производится вызов функции faktor и передается ее аргумент i

При описании функции типу ее результата присваивается тип длинный целый (Long), т.к., например, $10!=40320$, что выходит за диапазон типа Integer. При выполнении функции результат присваивается ее имени

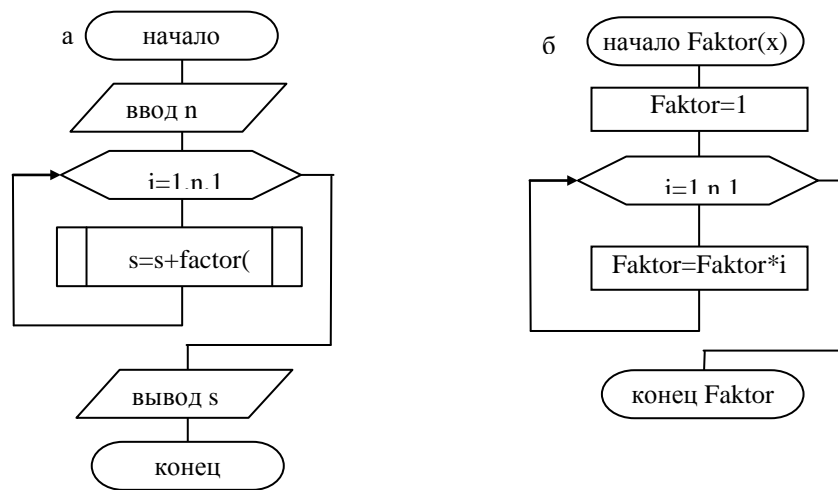


Рисунок 12 - Блок-схема программы progб (а) и подпрограммы Faktor (б)