Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
**«Кузбасский государственный технический университет
имени Т. Ф. Горбачева»**

Кафедра прикладных информационных технологий

Составитель
**Л. С. Таганов**

# ПРОГРАММИРОВАНИЕ ЗАДАЧ СРЕДСТВАМИ VBA

## Лабораторный практикум

Рекомендовано учебно-методической комиссией специальности
130400.65 «Горное дело» в качестве электронного издания
для использования в учебном процессе

Кемерово 2013

## РЕЦЕНЗЕНТЫ

Соколов И. А., доцент, к.т.н, заведующий кафедрой прикладных информационных технологий

Удовицкий В. И., профессор, д.т.н., председатель учебно-методической комиссии специальности 130400.65 «Горное дело»

**Таганов Леонид Степанович**.

**Программирование задач средствами VBA:** лабораторный практикум по дисциплине «Информатика» [Электронный ресурс] для студентов специальности 130400.65 «Горное дело» очной формы обучения / сост. Л. С. Таганов. – Электрон. текстовые дан. (124 Кб). – Кемерово: КузГТУ, 2013. – 1 электрон. опт. диск (CD-ROM); 12 см. – Систем. требования: Pentium IV; ОЗУ 256 МБ; Windows 7, 8; CD-ROM-дисковод; мышь. – Загл. с экрана.

В методических указаниях изложены уточненные теоретические основы, практические рекомендации и обновленные варианты заданий по выполнению лабораторных работ. Комплекс рассчитан на 20 часов лабораторных занятий.

# Оглавление

# 1. Программирование задач средствами VBA

## 1.1. Основные понятия языка VBA Excel

VBA (Visual Basic for Applications) относится к языкам объектно-ориентированного программирования, поэтому знакомство с ним естественно начать с понятия объекта.

### 1.1.1 Объекты

Объект – основной элемент VBA Excel. В VBA объектами являются рабочая книга, рабочий лист и его составляющие.
Примеры объектов:
Sheet – лист Excel;
Cell – ячейка;
Range – диапазон ячеек;
Application – приложение;
UserForm – пользовательская форма.
Доступ к объекту возможен через его методы и свойства.

### 1.1.2. Методы

Над объектами можно совершать различные действия. Действия, совершаемые над объектами, называются методами. Например, ячейку можно очистить (Clear), приложение закрыть (Quit), пользовательскую форму показать (Show) или скрыть (Hide). Название метода от названия объекта отделяется точкой: объект.метод.
Примеры использования методов:
Range(“B2:E2”).Select – выбрать диапазон ячеек B2:E2;
Range(“C1:C5”).Clear – очистить диапазон ячеек C1:C5;
UserForm2.Hide – скрыть форму № 2;
UserForm5.Show – показать форму № 5;
Application.Quit – выйти из приложения

### 1.1.3. Свойства

Свойствами описываются характеристики объектов. Например, размер и цвет шрифта, положение формы на экране или состояние объекта (доступность, видимость). Чтобы изменить характеристику объекта, надо просто изменить значение свойства, т.е. присвоить ему определенные значения.

Синтаксис установки значения свойства:

Объект. Свойство = Значение Свойства.

Объект обозначает имя объекта, Свойство – имя свойства, которому присваивается значение. Имя объекта отделяется от имени свойства точкой.

Примеры свойств:

Range(“D1”).Value = 2005 – поместить в ячейку D1 значение 2005.

Range(“C1:C10”).Text = “Информатика” – поместить в диапазон ячеек C1:C10 текст Информатика.

Range(“B2”).Font.Size = 14 – в ячейке B2 установить размер шрифта 14.

### 1.1.4. Методы и свойства некоторых объектов VBA

Объект: Application – приложение.

Метод: Quit – закрыть.

Свойство: Caption – имя главного окна.

Примеры. Application.Quit – закрыть приложение; Application.Caption = “Протокол” – установить в качестве заголовка окна приложения “Протокол”.

Объект: Sheet (лист), ActiveSheet (активный/выбранный лист).

Семейство: Sheets (листы).

Методы: Select – выбрать:

ShowDataForm – показать встроенную форму.

Примеры: Sheets(“Меню”).Select – выбрать лист “Меню”;

ActiveSheet.ShowDataForm – на активном в настоящий момент листе показать встроенную форму.

Объект: Range – диапазон ячеек.

Метод: Clear – очистить все ячейки рабочего листа от текста.

Свойство: Name – имя.
Примеры. Sheets(Протокол»).Range(“B4:B10”).Name = “Класс” – диапазону B4:B10, расположенному на листе “Протокол”, присвоить имя “Класс”;
Sheets(“Протокол”).Range(“B4:B10”).Select – выделить диапазон B4:B10 на листе “Протокол”.
Объект, Семейство: UserForm – пользовательская форма.
Методы: Show – показать; Hide – скрыть.
Свойство: Caption – текст, отображаемый в строке заголовка.
Примеры. UserForm1.Show – показать пользовательскую форму номер один;
UserForm1.Hide – скрыть пользовательскую форму номер один;
UserForm1.Caption – “Информатика” – вывести в строке заголовка заданный в кавычках текст.
Объект, Семейство: TextBox (Поле ввода).
Свойство: Text (содержимое).
Примеры. UserForm1.TextBox1.Text = Date – в поле ввода номер один в пользовательской форме номер один записать текущую дату;
UserForm1.TextBox2.Text = “” – очистить поле ввода номер два в пользовательской форме номер один.
Объект, Семейство: ComboBox – поле со списком.
Метод: AddItem – добавить элемент в список.
Свойства: Text (содержимое); Rowsource (источник строк для списка).
Примеры. UserForm1.ComboBox2.Text = “” – очистить значение поля для поля ввода со списком номер два в пользовательской форме номер один; UserForm2.ComboBox1.Rowsource= “B2:B10” – источником строк для поля один со списком в пользовательской форме два установить данные из диапазона B2:B10;
UserForm1.ComboBox1.AddItem (“Факс”) – добавить к списку элемент, заключенный в кавычки.
Объект, Семейство: OptionButton.
Свойства: Value – значение; Caption – надпись.
Примеры. UserForm3.OptionButton1.Value = True – выбрать переключатель номер один в пользовательской форме номер три;
UserForm3.OptionButton1.Capture = “Успеваемость” – установить

надпись “Успеваемость” рядом с переключателем в пользовательской форме номер три.
Объект, Семейство: CheckBox.
Свойства: Value – значение; Caption – надпись.
Примеры. UserForm2.CheckBox1.Value = True – установить флажок номер один в пользовательской форме номер два; UserForm3.CheckBox1.Value = False – сбросить флажок номер один в пользовательской форме номер три; UserForm4.CheckBox1.Capture = “Класс” – установить надпись “Класс” рядом с флажком в пользовательской форме номер четыре.

### 1.1.5. События

Событие представляет собой действие, распознаваемое объектом (например, щелчок мышью или нажатие клавиши, перемещение мыши или выход из программы), для которого можно запрограммировать отклик, т.е. реакцию объекта на произошедшее событие.

В языке VBA для каждого объекта определен набор стандартных событий. Стандартное событие для объекта “кнопка” (CommandButton) – щелчок мышью (Click).

Если пользователь нажимает на кнопку, то это событие. На это событие должен быть отклик, т.е. выполнение какой-либо программы. Такая программа называется процедурой обработки событий и имеет стандартное имя. Если такой отклик не создан (не написана соответствующая программа), то система никак не будет реагировать на это событие.

### 1.1.6. Элементы языка VBA

Объекты – основные элементы языка VBA, но не единственные. К другим элементам относятся: константы, переменные, массивы, выражения, встроенные функции, встроенные диалоговые окна, операторы.

Константы – данные, не изменяющиеся в процессе решения задачи. Константы бывают двух видов: числовые и символьные.
Числовые константы – это целые либо вещественные числа.

Символьные константы – текст, заключенный в кавычки. Пример числовой константы – 5,8 (использование запятой или точки зависит от настроек операционной системы). Пример символьной константы – ООО “Темп”.

Переменные – данные, значения которых меняются в ходе выполнения программы. Для переменной задается имя и тип.

Имя переменной должно начинаться с буквы, и может содержать любую комбинацию букв, цифр и символов за исключением точек, пробелов и следующих символов: “!”, “%”, “&”, “$”, “#”, “@”. Длина имени не должна превышать 255 символов. Не следует использовать имена, совпадающие с ключевыми словами VBA и именами встроенных функций и процедур.

Основные типы переменных, их размеры и диапазоны принимаемых значений приведены в табл. 1.

Таблица 1

| Тип | Размер (байт) | Диапазон значений |
|---|---|---|
| Byte (байт) | 1 | От 0 до 255 |
| Boolean (логический) | 2 | True или False |
| Integer (целое число) | 2 | От -32 768 до 32 767 |
| Long (длинное целое число) | 4 | От -2 147 483 648 до 2 147 483 647 |
| Single (число с плавающей запятой обычной точности) | 4 | Для отрицательных значений от -3,4E38 до -1,4E-45. Для положительных – от 1,4E-45 до 3,4E38 |
| Double (число с плавающей запятой двойной точности) | 8 | Для отрицательных от -1,79E308 до -4,94E-342. Для положительных – от 4,94E-324 до 1,79E308 |
| Currency (денежный) | 8 | От 922 337 203 685 477,5808 до 922 337 203 685 477,5807 |
| Date (дата и время) | 8 | От 1 января 100 г. до 31 декабря 9999 г. |
| String (строка) | 10+длина строки | От 0 до $2\times 10^9$ |
| Variant (вариант) | | Зависит от содержимого переменной |

Переменные типа Variant могут хранить все, что в них поместят. Их тип изменяется в зависимости от последнего присвоения. В программах переменные описываются с помощью специального оператора Dim.

Массив – упорядоченная совокупность однотипных переменных. Массивы имеют имя и размерность. Имя массива подбирается с учетом тех же правил, что и имена переменных. Размерность – это количество элементов (переменных), составляющих массив.

Из констант, переменных и встроенных функций (они рассмотрены далее) с помощью скобок и знаков арифметических операций (“+”, “-”, “*”, “/”, “^”) можно составлять выражения. Частным случаем выражения может быть просто одиночный элемент, т.е. константа, переменная или обращение к встроенной функции.

В VBA имеется большой набор встроенных функций, которые разделяют на категории

С помощью объекта Applications можно вызвать более 400 встроенных функций рабочего листа, используя конструкцию вида:
Application.Функция Рабочего Листа(Аргументы).
Примеры:
Application.Sum(Sheets(“Проверка”).Range(“A1:B20”)) – суммируются значения из ячеек диапазона A1:B20, расположенного на листе “Проверка”;
Application.CountA(Sheets(“Ученики”).Range(“A:A”)) – подсчитывается количество непустых ячеек в столбце A на листе “Ученики”.

### 1.1.7. Ввод и вывод с помощью стандартных окон

В Visual Basic for Applications имеются стандартные окна для ввода данных в программу и отображения результатов, т.е. вывода данных.

Окно ввода InputBox позволяет организовать простейший диалог с пользователем.
Синтаксис окна ввода следующий: Q = InputBox(“текст”, ”заголовок окна”)

Этот оператор выводит на экран диалоговое окно, содержащее текст сообщения и поле для ввода данных. Ожидает ввода данных пользователем и нажатия кнопки. Введенное значение записывается в переменную Q.
Например, C= InputBox(“Введите стоимость изделия C=”, “Запрос на ввод стоимости”).

Окно сообщений MsgBox может вызываться как процедура (только для вывода) или как функция (для вывода и принятия ответа пользователя).
Синтаксис вызова процедуры следующий: MsgBox (сообщение, кнопки + значки, заголовок окна).

Функция MsgBox выводит на экран диалоговое окно, содержащее сообщение, ожидает нажатия кнопки пользователем и возвращает значение типа Integer, указывающее, какая кнопка была нажата.
Синтаксис вызова функции следующий:
кнопка=MsgBox(сообщение, кнопки+значки, заголовок окна).
Параметр сообщение содержит текст, который выводится в окне диалога.
Параметр кнопки указывает число и тип отображаемых кнопок в окне сообщения. Возможные значения аргумента:

Константы, задающие количество кнопок

| Константа | Значение | Отображаются кнопки |
|---|---|---|
| vbOKOnly | 0 | OK |
| vbOKCancel | 1 | OK, Отмена |
| vbYesNoCancel | 3 | Да, Нет, Отмена |
| vbYesNo | 4 | Да, Нет |
| vbRetryCancel | 5 | Повтор, Отмена |

Константы, соответствующие значениям функции MsgBox

| Константа | Значение | Нажатая кнопка |
|---|---|---|
| vbOK | 1 | OK |
| vbCancel | 2 | Отмена |
| vbAbort | 3 | Прервать |

| | | |
|---|---|---|
| vbRetry | 4 | Повторить |
| vbIgnore | 5 | Пропустить |
| vbYes | 6 | Да |
| vbNo | 7 | Нет |

### 1.1.8. Рекомендации по минимизации количества ошибок при составлении кода программы на VBA

Любой программист должен стремиться составить надежную программу. Надежная программа – это программа, которую легко понять и она может работать без особых проблем.

Составление надежной программы – результат больших усилий и опыта, что потребует много времени и упражнений. Для облегчения такой работы направлены некоторые рекомендации:

- обязательно объявлять переменные явно, для этого в начале каждого программного модуля включать инструкцию *Option Explicit* (все переменные объявлять явно). Это простое действие, возможно, является самым важным в написании надежной программы, потому что предотвращает все досадные ошибки и дефекты, которые происходят из-за неправильно написанных названий переменных, и их, как правило, так тяжело выявить;
- не использовать без крайней необходимости тип *Variant*, так как он иногда является причиной некоторых трудноуловимых ошибок в программе;
- не использовать без крайней необходимости тип *Object*, так как он может содержать ссылку на любой тип объектов, а эта гибкость может быть источником проблем. При всякой возможности объектные переменные объявлять как конкретный тип, на который они будут ссылаться, например, *Excel.Application*;
- все объявления переменных помещать в начало процедуры (модуля) и располагать их по одному объявлению на строку;
- не писать в одной строке несколько операторов, даже если возможно поместить в одной строке несколько операторов через двоеточие, потому что такую программу будет затруднительно читать;

- проверять корректность данных, получаемых процедурами в качестве аргументов и введенных пользователями. Прежде чем использовать данные в программе их необходимо проверять, для этого целесообразно использовать, например, функции проверки типов (IsNumeric, IsArray и другие).

Следуя этим рекомендациям можно гарантированно сократить время разработки и отладки программы.

### 1.1.9. Возможные типы ошибок

Все возникающие в программе ошибки можно разделить на три типа:

- ошибки выполнения (*runtime errors*), возникающие в процессе выполнения программы. Если не предусмотрен механизм их перехвата, то происходит аварийная остановка программы с выдачей стандартного описания такой ошибки;
- логические ошибки (*bugs*), приводящие к неправильной работе программы, выражающиеся в получении неправильных результатов. Ошибки такого типа не приводят к аварийной остановке программы. Устранение таких ошибок возможно лишь исправлением алгоритма программы, который должен разрабатываться программистом на одном из первых этапов решения поставленной задачи;
- синтаксические ошибки (*syntax errors*), возникающие в процессе набора текста программы при нарушении правил синтаксиса языка VBA. Каждая строка текста программы в процессе ее набора анализируется редактором. Рекомендуется набирать текст программы в нижнем регистре, тогда после перехода на следующую строку редактора введенный текст изменится в соответствии с синтаксисом языка VBA (некоторые буквы отобразятся в верхнем регистре). Если этого не произошло, необходимо проверить набранную строку программы.

### 1.1.10. Структура редактора VBA

Редактор VBA активизируется командой Сервис Макрос Редактор VBA или комбинацией клавиш Alt + F11. После выполнения команд мы попадаем в редактор VBA. Возвратиться из редактора VBA в рабочую книгу можно нажатием кнопки Вид(View) Microsoft Excel.

### 1.1.11. Основные компоненты окна редактора

Окно редактора состоит из следующих компонентов:

- меню;
- панели инструментов;
- окно проекта;
- окно свойств;
- окно редактирования кода.

Вид окна редактора представлен на рисунке 1

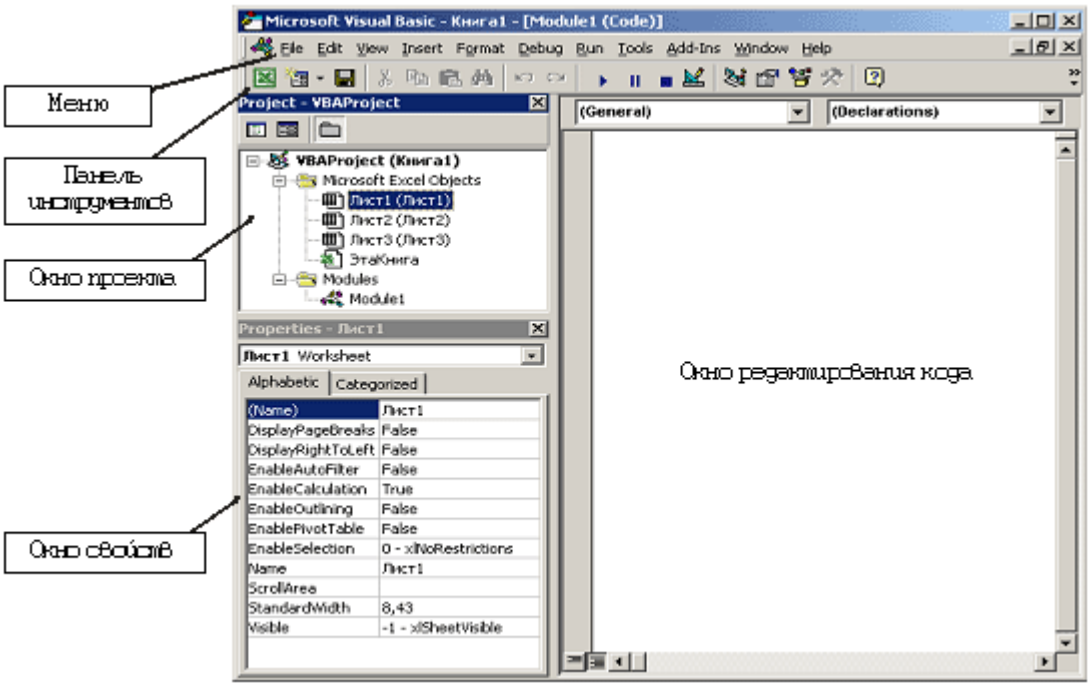


Рис 1

### 1.1.12. Панели инструментов

Стандартная панель инструментов редактора Visual Basic содержит кнопки, предназначенные для выполнения наиболее часто используемых команд. Панель разбита на отдельные сегменты по типу выполняемых команд (рисунок 2).

Рис 2

Кнопки первого сегмента (рисунок 3):

- для возврата в Excel;
- вставки элементов (модулей, процедур, экранных форм) в проект;
- сохранение рабочей книги.

Рис 3

Кнопки второго сегмента (рисунок 4):

- для вырезания;
- копирования;
- вставки;
- поиска фрагментов кода.

Рис 4

Кнопки третьего сегмента (рисунок 5):

- отмены действий;
- повторения отмененных действий.

Рис 5

Кнопки четвертого сегмента (рисунок 6):

- для выполнения процедуры;
- остановки выполнения процедуры;

- прекращения выполнения процедуры;
- смены режима отображения экранной формы.

Рис 6

Кнопки пятого сегмента рисунок 7:

- управляющие отображением окон проектов;
- управляющие отображением свойств;
- управляющие отображением просмотра объектов;
- управляющие отображением панели инструментов.

Рис 7

Последняя кнопка – это обычная кнопка вызова справочной системы (рисунок 8).

Рис 8

### 1.1.13. Окно проекта

Окно проекта активизируется выбором команды *Вид*/*Окно проекта* (*View, Project window*) или нажатием кнопки *Окно проекта.*

В окне проекта представлена иерархическая структура файлов, форм и модулей текущего проекта (рисунок 9).

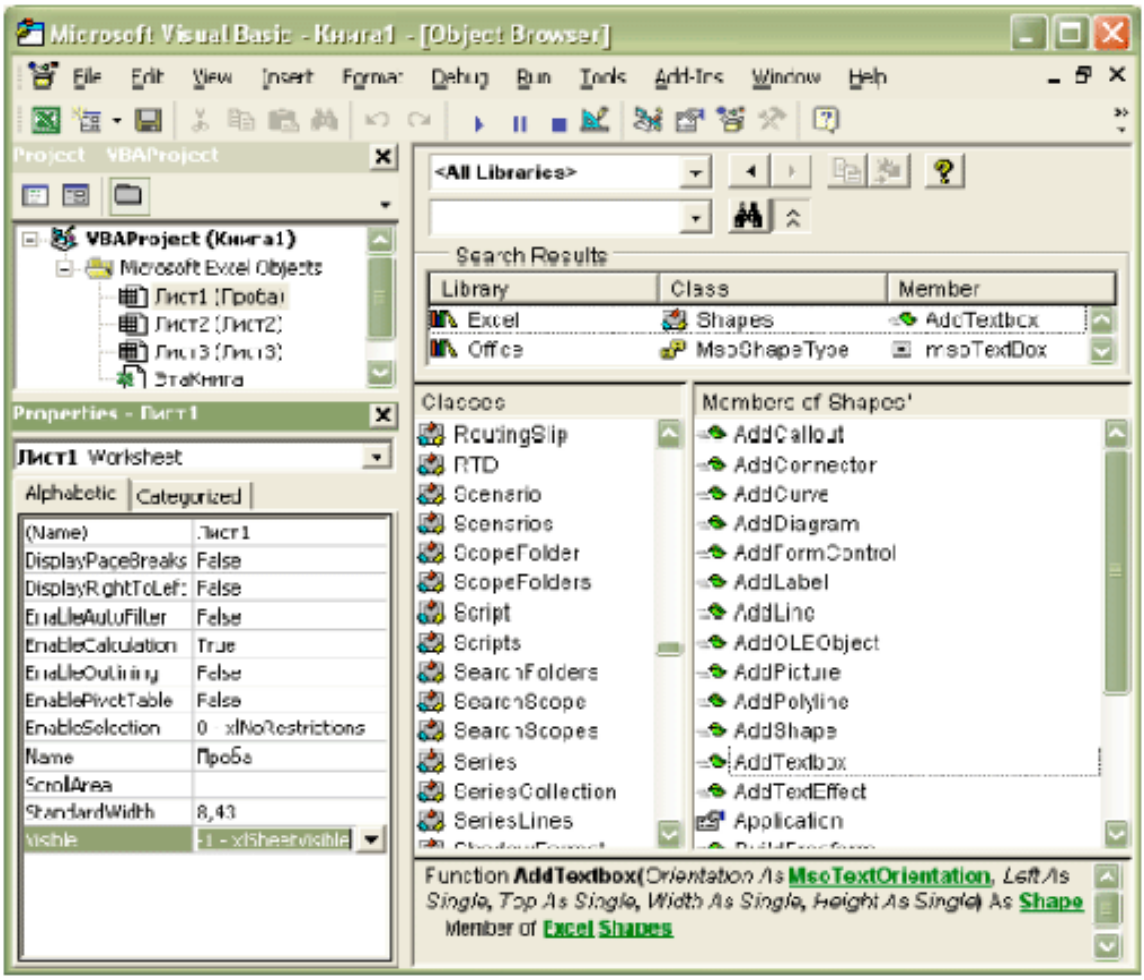


Рис 9

## 1.1.14. Окно свойств

В окне свойств перечисляются основные значения свойств выбранного объекта. Используя это окно, можно просматривать свойства и изменить их значения. Для просмотра свойств выбранного объекта надо выполнить команду *Вид/Окно свойств* (*Properties Windows*) или нажатием кнопки.

Окно свойств состоит из двух составных частей: верхней и рабочей.

В верхней части окна располагается раскрывающийся список, из которого можно выбрать любой элемент управления текущей формы или саму форму. Рабочая часть состоит из двух вкладок, в которых свойства располагаются по алфавиту (Alphabetic) и по категориям (Categorized) (рисунок10).

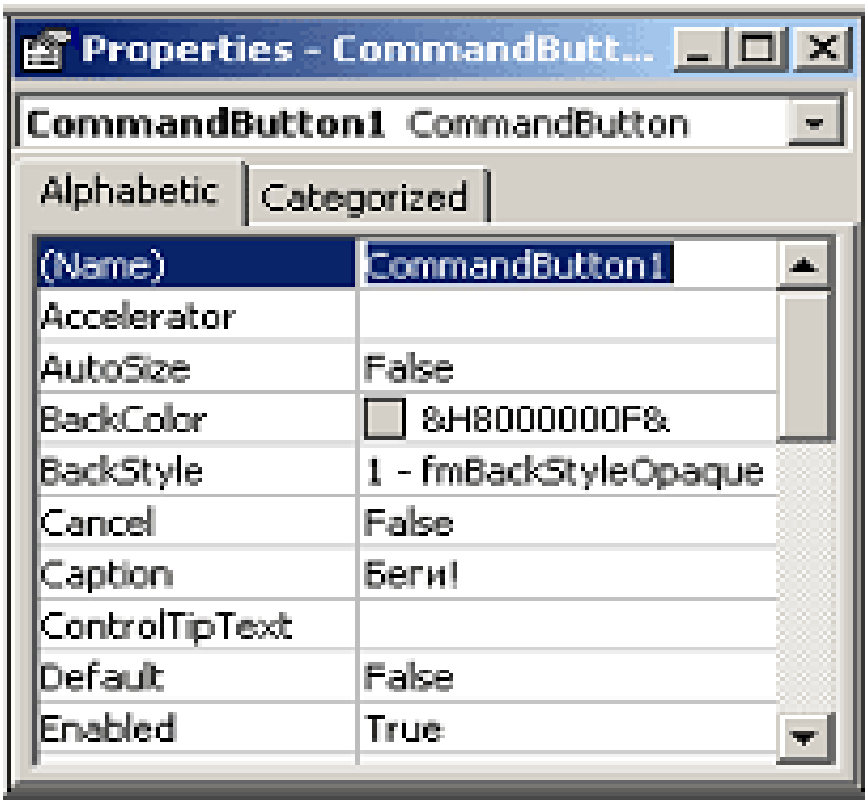


Рис 10

### 1.1.15. Окно для просмотра объектов (Object Browser)

Окно *Просмотр объектов* (*Object Browser*) вызывается командой *Вид/Просмотр объектов* (*View, Object Browser*) или нажатием кнопки. В этом окне можно просматривать все объекты проекта. Здесь вы найдете все свойства, методы и события, связанные с любым объектом.

Окно *Просмотр объектов* состоит из трех основных частей:

1. Раскрывающегося списка Проект/Библиотека в верхнем левом углу экрана. Например, библиотеки объектов Excel, VBA, Office и VBAProject (объекты пользовательского проекта).
2. Списка Классы. Выводятся все классы выбранной библиотеки.
3. Списка Компоненты (Members). Выводятся все компоненты выбранного класса.

Это окно предоставляет доступ ко всем объектам, свойствам, методам и событиям (рисунок 9).

# 2. Лабораторная работа № 1. Практика набора, тестирования и отладки программы по образцу.

Открыть свою рабочую книгу MS Excel, созданную ранее. Если книга не была создана, тогда создать новую. Открыть редактор VBA и ввести текст программы. Каждый текст программы вводить на очередном свободном листе окна редактора, номер которого должен соответствовать номеру рабочего листа книги. Выполнить тестирование текста программы и при необходимости произвести ее отладку.

## 2.2.1. Образцы текстов программ

(Время на выполнение 4 часа)

### Технология выполнения лабораторной работы

1. В окно редактора VBA ввести программный код предложенных программ (поочередно).
2. Протестировать программу с разными исходными данными.
3. В случае возникновения ошибок отладить программу до ее безошибочного выполнения.

### Вопросы для защиты работы

1. Как открыть окно редактора VBA?
2. Почему рекомендуется применять команду Option Explicit?
3. При вводе текста программы появляется красная строка (или часть) теста. О чем это свидетельствует?
4. Почему рекомендуется набирать текст программы в нижнем регистре?
5. Если после перехода на новую стоку не произошло преобразования символов в заглавные в соответствии с синтаксисом, то о чем это свидетельствует и что необходимо выполнить?.
6. Как преобразуется текст исходной программы после ее тестирования и отладки в случае необходимости?

## Образец текста программы №1

```
Option Explicit ‘Переменные объявлять явно
Sub qwerty1()
Dim x As Byte
Dim y As Byte
Dim z As Byte
Dim m As Byte
Worksheets(“Лист1”).Select ‘Выбрать рабочий лист1 из семейства
                                рабочих листов
Cells.Clear ‘Очистить все ячейки рабочего листа
Randomize
x = Int(Rnd() * 50)
Randomize
y = Int(Rnd() * 50)
Randomize
z = Int(Rnd() * 50)
If x < y Then
   If x < z Then
        m = x
   Else
        m = z
   End If
ElseIf y < z Then
   m = y
Else
   m = z
End If
MsgBox (“x=”) & Cbyte(x)
MsgBox (“y=”) & Cbyte(y)
MsgBox (“z=”) & Cbyte(z)
MsgBox (“m=”) & Cbyte(m)
Cells(2, 2).Value = (“x=”) & Cbyte(x)
Cells(3, 2).Value = (“y=”) & Cbyte(y)
Cells(4, 2).Value = (“z=”) & Cbyte(z)
Cells(5, 2).Value = (“m=”) & Cbyte(m)
```

```
End Sub
```

## Образец текста программы №2

```
Option Explicit
Sub qwerty2()
Dim r As Integer
Dim n As Byte
Dim I As Byte
Dim j As Byte
Dim x() As Integer
Dim p As Variant
Worksheets("Лист2").Select
Cells.Clear
r = 0
I = 1
Do
    p = InputBox(«Введи n=«;"Размерность массива")
    If Not IsNumeric(p) Then MsgBox ("Повтори ввод")
Loop Until IsNumeric(p)
n = p
ReDim x(1 to n) As Integer
Do While I <= n
     x(i) = Fix(Sin(i) * 45)
     I = I + 1
Loop
For I = 1 to n
     Cells(2 + I, 4) = x(i)
Next i
j = 2
Do While j <= n
  If x(j) < 0 Then
     r = r + x(j)
  End If
  j = j + 2
Loop
```

```
Cells(3, 7) = r
End Sub
```

## 3. Лабораторная работа № 2. Создание, тестирование, отладка, оценка и анализ полученного результата программы с применением операторов ветвления

Цель работы: научиться разрабатывать разветвляющиеся алгоритмы, используя условный оператор IF.

### 3.1. Операторы альтернативы (ветвления)

Как и в любом другом языке программирования, в VBA можно проверять условия и выполнять действия в соответствии с результатами проверки этих условий. Для данной цели применяются следующие операторы (инструкции) принятия решения, позволяющие организовать в программе ветвление.

Условный оператор: *IF <условие> THEN <оператор (код)>*

Такая языковая конструкция позволяет выполнить один или несколько операторов в случае истинности проверяемого условия. Применяется однострочный или блочный вариант записи условного оператора. Если необходимо выполнить более одной строки кода, нужно использовать блочный вариант с ключевым словом End IF. С помощью такой инструкции реализуется базовый алгоритм неполного ветвления.

Синтаксис:

```
IF <условие> Then <оператор (код)>
IF <условие> Then
    <блок кода>
End IF
```

---

Примеры:

```
IF x<10 Then z=0

IF x>10 Then
    z=2
```

```
    z=z + x
End IF
```

Инструкция, реализующая базовый алгоритм полного ветвления, позволяет определить два блока операторов. Первый выполняется, когда условие истинно, а второй, когда оно ложно.

Пример:

```
IF x<>0 Then
    y=Sin(x)/x
Else
    y=1
End IF
```

---

Пример ветвления по четырем направлениям:

```
IF <условие 1> Then
    <блок кода 1>
ElseIF <условие 2> Then
    <блок кода 2>
ElseIF <условие 3> Then
    <блок кода 3>
Else <блок кода 4>
End IF
```

В блоке IF допускается любое количество предложений ElseIF, но ни одно из них не может находиться после предложения Else. Однако с точки зрения методологии структурного программирования уровень вложенности оператора IF не должен превышать трех.

Пример:

```
IF x=-1.57 Then
    y=-1
ElseIF x=0 Then
    y=0
ElseIF x=1.57 Then
    y=1
Else y=Sin(x)
End IF
```

---

Пример 1. Вычислить арифметическое выражение y = x + 1, если x < 0, либо y = 2 *x, если x <> 0. Программа будет выглядеть так:

```
Sub Пример1()
Dim x As Integer, y As Integer
x = InputBox(«Введите число»)
If x < 0 Then y =x + 1 Else y = 2 *x 'Проверка условия
MsgBox(“Ответ: y =”) & Cint(y)
End Sub
```

---

Пример 2. С клавиатуры водятся два числа. Если a > b, то t1 = a^2, t2 = 2*(a + b). Иначе t1 = 2*a, t2 = a-b. Значения t1 и t2 вывести на экран.

```
Sub Пример2()
Dim a As Integer, b As Integer
Dim t1 As Integer, t2 Is Integer
a = InputBox(«Введите значение a») ‘Ввод исходных данных
b = InputBox(«Введите значение b») ‘Ввод исходных данных
If a > b Then
    t1 = a^2
    t2 = 2*(a + b)
Else
    t1 = 2*a
    t2 = a – b
End If
MsgBox(“t1=”) & Cint(t1)
MsgBox(“t2=”) & Cint(t2)
End Sub
```

## Технология выполнения лабораторной работы

1. Уяснить задачу и принять решение по ее выполнению.
2. Составить блок-схему решения задачи.

3. На основе блок-схемы составить код программы и ввести его в окно редактора VBA.
4. Протестировать программу и отладить ее при необходимости.
5. Результаты решения должны соответствовать поставленной задаче.

## Вопросы для защиты работы

1. Какая конструкция у оператора повтора?
2. Какие операторы разветвления Вы знаете?
3. Чем заканчивается оператор IF, если в ветке «да» несколько операторов?
4. Как выглядит полная форма оператора IF?
5. Какие выражения применяются при использовании оператора ветвления?

## 3.2. Задания для выполнения лабораторной работы

(Время на выполнение 2 часа)

(Вариант выбирается в соответствии с номером рабочего места)

1. Даны три числа. Эти числа заранее неизвестны. Найти максимальное и минимальное из этих чисел.
2. Даны два числа. Эти числа заранее неизвестны. Вывести на экран первое число, если оно больше второго, и оба, если это не так.
3. Даны два числа. Эти числа заранее неизвестны. Меньшее из этих чисел заменить их удвоенным произведением, а большее – их полу-суммой.
4. Даны два числа. Эти числа заранее неизвестны. Заменить второе число нулём, если оно не меньше первого, и оставить его прежним, если это не так. Первое число оставить без изменения.
5. Даны четыре числа. Эти числа заранее неизвестны. Отрицательные числа возвести в квадрат.
6. Даны три числа. Эти числа заранее неизвестны. Найти максимальное из них и разделить его на разность двух других.
7. Даны три числа. Эти числа заранее неизвестны. Найти минимальное из них и умножить его на полу-сумму двух других.

8. Даны четыре числа. Эти числа заранее неизвестны. Найти максимальное из них и разделить его на разность других.
9. Даны четыре числа. Эти числа заранее неизвестны. Найти из них максимальное и минимальное.
10. Даны четыре числа A, B, C и D. Определить число среди первых трех, которое равно D. Если это не так, то D присвоить A+B+C.
11. Даны четыре числа. Эти числа заранее неизвестны. Найти максимальное, минимальное числа и их сумму.
12. Даны четыре числа A, B, C и D. Найти максимум из D-A, D-B и D-C.

Примечание. 1. При выполнении задания предусмотреть возможное (допустимое) оформление диалоговых окон ввода данных и вывода результатов.
2. Вывод результатов реализовать в диалоговом окне вывода и на рабочий лист Excel.

## 4. Лабораторная работа № 3. Создание, тестирование, отладка, оценка и анализ полученного результата программы с применением оператора выбора варианта

### 4.1. Оператор выбора варианта

Цель работы: изучить принцип действия оператора выбора Select.

При выборе для выполнения одного из нескольких операторов (блоков операторов) целесообразно и удобно использовать инструкцию Select Case. С помощью этого оператора в языке реализована алгоритмическая конструкция множественного выбора.

Оператор выбора действует следующим образом. Если значение выражения равно одной из констант, то выполняется соответствующий ей оператор. Затем управление передаётся за пределы оператора выбора.

Если значение выражения не совпадает ни с одной константой, то управление передаётся оператору, стоящему после Case Else.

Синтаксис:

```
Select Case<переменная или выражение>
        Case <значение 1>
            <оператор (блок операторов) 1>
        Case <значение 2>
            <оператор (блок операторов) 2>
        Case <значение 3>
            <оператор (блок операторов) 3>
End Select
```

Пример1. Программа запрашивает порядковый номер дня недели и выдаёт название этого дня недели.

```
Sub Дни_недели()
Dim a As Byte
a = InputBox(«Введите номер дня недели”; “Параметр выбора”)
Select Case a
Case 1: MsgBox(“Понедельник”)
Case 2: MsgBox(“Вторник”)
Case 3: MsgBox(“Среда”)
Case 4: MsgBox(“Четверг”)
Case 5: MsgBox(“Пятница”)
Case 6: MsgBox (“Суббота”)
Case 7: MsgBox(“Воскресенье”)
Case Else: MsgBox(«Такого номера не существует»)
End Select
End Sub
```

---

Пример 2. Использование оператора выбора варианта в подпрограмме-функции:

```
Function PR(ByVal S As Single, ByVal P As Integer) As Single
    Select Case P
        Case 0
            PR=S*0
        Case 1
            PR=S*0.10
        Case 2
            PR=S*0.15
```

```
            Case 3
                    PR=S*0.20
    End Select
End Function
```

---

Пример3. Программа, вызывающая подпрограмму функцию:

```
Sub Krb()
Dim S As Single
Dim Sum As Single
Dim P As Integer
Dim Prom As Variant
S = 5000
 Do
    Prom=InputBox("Введите номер варианта <от 0 до 5>")
    IF Not IsNumeric(Prom) Then MsgBox("Повторите ввод!")
 Loop Until IsNumeric(Prom)
 P=Prom
 Sum=PR(S,P) 'Вызов процедуры-функции. Оператор вызова PR(S,P)
 MsgBox("Значение S=") & CSng(S)
End Sub
```

Допускается вложенность инструкций Select Case. При этом каждой вложенной инструкции Select Case должна соответствовать инструкция End Select.

## Технология выполнения лабораторной работы

1. Уяснить задачу и принять решение по ее выполнению.
2. На основе блок-схемы составить код программы и ввести его в окно редактора VBA.
3. Протестировать программу и отладить ее при необходимости.
4. Результаты решения должны соответствовать поставленной задаче.

## Вопросы для защиты работы

1. Какая конструкция у оператора выбора варианта?
2. Для чего предусмотрен оператор выбора варианта?

3. Чем заканчивается оператор выбора варианта?
4. Как выглядит полная форма оператора выбора варианта?
5. Какой тип переменной необходимо использовать при использовании оператора выбора варианта?

## 4.2. Задания для выполнения лабораторной работы

(Время на выполнение 2 часа)

1. Основания равнобедренной трапеции равны a и b, а бедро c. Написать программу, которая по заданному номеру варианта вычисляет значения: высота, периметр и площадь.
2. Диагонали ромба равны a и b. Написать программу, которая по заданному номеру варианта вычисляет значения: длина стороны, периметр и площадь.
3. Площадь круга равна S. Написать программу, которая по заданному номеру варианта вычисляет значения: радиус, диаметр и длина окружности.
4. Стороны прямоугольника равны a и b. Написать программу, которая вычисляет по заданному номеру варианта значения периметр, площадь и длина диагонали.
5. Сторона куба равна a. Написать программу, которая по заданному номеру варианта вычисляет значения: объём, площадь боковой поверхности и периметр.
6. Катеты прямоугольного треугольника равны a и b. Написать программу, которая по заданному номеру варианта вычисляет значения: диагональ, площадь и периметр.
7. Сторона равностороннего треугольника равна a. Написать программу, которая по заданному номеру варианта вычисляет значения: периметр, площадь и, высота.
8. Основание параллелограмма a, боковая сторона b. Написать программу, которая по заданному номеру варианта вычисляет значения: периметр, площадь и, высота.
9. Объем куба равен V. Написать программу, которая по заданному номеру варианта вычисляет значения: длину ребра, площадь боковой поверхности и периметр.

10. Площадь квадрата равна S. Написать программу, которая по заданному номеру варианта вычисляет значения: длина стороны, диагональ и периметр.
11. Используя формулу Герона ($S=\sqrt{p(p-a)(p-b)(p-c)}$, где p = (a+b+c)/2), Написать программу вычисления площади треугольника при значениях сторон: 1) a=10, b=6 и c=8; 2) a=8, b=10 и c=6; 3) a=6, b=4 и c=4.
12. Радиус круга равна R. Написать программу, которая по заданному номеру варианта вычисляет значения: диаметр, длина окружности и площадь.
13. Используя формулу Герона ($S=\sqrt{p(p-a)(p-b)(p-c)}$, где p = (a+b+c)/2), Написать программу вычисления площади треугольника при значениях сторон: 1) a=8, b=6 и c=4; 2) a=12, b=10 и c=8; 3) a=6, b=4 и c=8.

# 5. Лабораторная работа № 4. Создание, тестирование, отладка, оценка и анализ полученного результата программы с применением операторов цикла

## 5.1. Операторы циклов

DO … LOOP

С помощью такой инструкции в языке реализована базовая алгоритмическая конструкция повторения, и она позволяет многократно выполнить любой блок кода.

Существует несколько вариантов записи этого оператора, но в каждом из них проверяется условие, управляющее работой цикла, и по результатам проверки определяется необходимость его продолжения. Результатом вычисления логического выражения (управляющего условия) будет одна из констант True или False.

### 5.1.1. Циклы с предусловием

DO WHILE <условие> … LOOP

Оператор DO WHILE <условие> … LOOP позволяет проверить условие перед началом цикла и выполнять цикл пока оно имеет

значение True. Как только условие, управляющее работой цикла, примет значение False, выполнение тела цикла прекратится.

Пример:

```
Dim X As Integer ‘Описание переменной X целого типа
X=0 ‘Начальное значение переменной X
DO WHILE X<=10 ‘Пока X<=10, цикл будет повторяться
    X=X+1 ‘Изменение значения переменной X
LOOP ‘Конец цикла
```

Другой вариант записи инструкции такого цикла:

WHILE <условие>… WEND

---

Пример:

```
X=0
WHILE X<12
    Y=Cos(X)
    X=X+1
WEND
```

DO UNTIL <условие> … LOOP

Оператор Do Until <условие> … Loop позволяет проверить условие перед началом цикла и выполнять тело цикла пока условие принимает значение False. Как только управляющее условие примет значение True, выполнение тела цикла прекратится.

Пример:

```
Dim X As Integer ‘Описание переменной X целого типа
X=0 ‘Начальное значение переменной X
Do Until X>10 ‘До тех пор, пока X<=10, цикл повторяется
    X=X+1 ‘Изменение значения переменной X
Loop ‘Конец цикла
```

### 5.1.2. Циклы с постусловием

DO … LOOP WHILE <условие>

Если операторы цикла необходимо выполнить хотя бы раз, то для этой цели нужно использовать цикл с постусловием. Инструкция Do … Loop While <условие> позволяет проверить условие после

выполнения операторов тела цикла. Цикл будет повторяться до тех пор, пока выражение в условии цикла имеет значение True. Как только условие цикла примет значение False, выполнение тела цикла прекратится.

Пример:

```
Dim X As Integer
X=0
Do
   X=X+1
Loop While X<=10 ‘До тех пор, пока X<=10, цикл повторяется
```

DO … LOOP UNTIL <условие>

В отличие от предыдущего этот цикл будет выполняться до тех пор, пока значение управляющего условия равно False.

Пример:

```
Dim X As Integer
X=0
Do
    X=X+1
Loop Until X>10 ‘Как только переменная станет больше десяти,
                выполнение цикла прекратится
```

### 5.1.3. Цикл по счетчику

FOR … NEXT

Цикл с определенным количеством повторений выполняется от начального до конечного значения параметра с заданным шагом.

Пример:

```
Dim X,S As Integer
S = 0
For X=1 To 10 ‘Повторять цикл для X, изменяющегося от 1 до 10 с
                шагом 1
  S=S + x
Next X ‘Конец цикла

Exit For или Exit Do ‘Досрочный выход из цикла
```

### 5.1.4. Вложенные циклы

Совокупность простых циклов, вложенных один в другой, называется сложным (вложенным) циклом. При конструировании сложных циклов необходимо руководствоваться следующими правилами:

- нельзя войти во внутренний цикл, минуя вход внешнего цикла;
- имена параметров простых циклов не должны повторяться в конструкции сложного цикла;
- простые циклы не должны пересекаться в конструкции сложного цикла, то есть окончание внешнего цикла не должно предшествовать окончанию внутреннего цикла.

Примеры:

```
For i=1 to n
    For j=1 to m
        A(I, j)=Int(Sin(j*i)*100)
    Next j
Next i
```

---

```
Do
   X=1
   Z=0
   Do
      S=Int(Rnd(x)*100)
      Z=Z+S
      X=X+1
   Loop Until X>=20
   Zsr=Z/20
Loop Until Zsr>=25
```

### Технология выполнения лабораторной работы

1. Уяснить задачу и принять решение по ее выполнению.
2. Составить блок-схему решения задачи.

3. На основе блок-схемы составить код программы и ввести его в окно редактора VBA.
4. Протестировать программу и отладить ее при необходимости.
5. Результаты решения должны соответствовать поставленной задаче.

## Вопросы для защиты работы

1. Какая конструкция у оператора цикла с предусловием?
2. Какая конструкция у оператора цикла с предусловием?
3. Какие операторы циклов Вы знаете?
4. Приведите пример вложенного (сложного) цикла?
5. Какие требования предъявляются при составлении сложных циклов?
6. Как выглядит полная форма оператора с параметром (по счетчику)?
7. Какие выражения применяются при использовании операторов цикла с предусловием и с постусловием?

## 5.2. Задания для выполнения лабораторной работы

(Время на выполнение 2 часа)

1. Найти сумму ряда чисел от 1 до 10, используя циклы: с предусловием (пока ложь), с постусловием (до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.
2. Найти сумму ряда чисел от 1 до 10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.
3. Найти сумму ряда чисел от 1 до 10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока ложь) и с параметром. При решении задания применить оператор выбора варианта Select Case.
4. Найти сумму чисел применив функцию Sin(x), 1<=x<=10, используя циклы: с предусловием (пока ложь), с постусловием

(до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.

5. Найти сумму чисел применив функцию Cos(x), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.
6. Найти сумму чисел применив функцию Sin(x)+Cos(x), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока ложь) и с параметром. При решении задания применить оператор выбора варианта Select Case.
7. Найти произведение чисел, применив функцию Sin(x), 1<=x<=10, используя циклы: с предусловием (пока ложь), с постусловием (до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.
8. Найти произведение чисел, применив функцию Cos(x), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока истина) и с параметром. При решении задания применить оператор выбора варианта Select Case.
9. Найти произведение чисел, применив функцию Sin(x)+Cos(x), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока ложь) и с параметром. При решении задания применить оператор выбора варианта Select Case.
10. Найти разность чисел, применив функцию Sin(x)+Cos(x), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока ложь) и с параметром. При решении задания применить оператор выбора варианта Select Case.
11. Найти сумму целых чисел применив функцию Int(Cos(x)*25), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока истина) и с параметром. При

решении задания применить оператор выбора варианта Select Case.

12. Найти сумму целых чисел применив функцию Fix((Sin(x)+Cos(x))*40), 1<=x<=10, используя циклы: с предусловием (пока истина), с постусловием (до тех пор, пока ложь) и с параметром. При решении задания применить оператор выбора варианта Select Case.

# 6. Лабораторные работы № 5, 6. Создание, тестирование, отладка, оценка и анализ полученного результата программы обработки массивов данных

## 6.1. Массивы

*Массив* – набор однотипных переменных с одним именем, каждая из которых называется элементом массива и имеет свой номер (индекс).

Массивы могут быть: одномерные (для нумерации элементов используется один индекс), двумерные (для нумерации элементов используются два индекса: номер строки, номер столбца) и *N*-мерные. Число измерений может достигать 60.

Кроме того, по способу выделения оперативной памяти для хранения элементов массивы подразделяются на статические и динамические.

### 6.1.1. Статические массивы

*Статическим* называется массив с заранее известным количеством элементов. Синтаксис описания (объявления) статического массива:

Dim <Имя массива>(<верхняя граница>) As <Тип>

По умолчанию значение нижней границы равно нулю**.**

Dim <Имя массива>(<Нижняя граница> To <Верхняя граница>) As <Тип>

Примеры:

Dim a(10) As Single ‘Одномерный массив с начальной границей, равной 0

Dim S(3 To 5) As String ‘Одномерный массив с явно заданными границами
Dim Z(1 To 3, 1 To 5) As Byte ‘Двумерный массив

Для задания по умолчанию нижней границы массива, равной 1, используется инструкции Option Base 1, которая задается в начале модуля.

Примеры:

```
Option Base 1
Sub Mas1()
Dim a(5) As Integer
Dim I As Integer, k As Integer
WorkSheets(“Лист1”).Select ‘Выбрать Лист1 из семейства листов
Cells.Clear ‘Очистить ячейки рабочего листа
K=2
For i=1 To 5
    a(i)=Int(Rnd(i)*100) ‘Формирование массива a(i)
    Cells(k , i+1)=a(i) ‘Вывод массива a(i) на рабочий лист.
Next i
End Sub
---------------------------------------------------------------------------------
Sub Mas2()
Dim a(1 to 5) As Integer, k As Integer
Dim I As Integer
WorkSheets(“Лист1”).Select
Cells.Clear
K=2
For i = 1 To 5 ‘Формирование одномерного массива a
     a(i)=Int(Rnd(i)*100)
Next i
For i=1 To 5 ‘Вывод элементов массива a(i) на рабочий лист.
    Cells(k , i+1)=a(i)
Next i
End Sub
-------------------------------------------------
Sub Dmas1()
```

```
Dim I As Integer
Dim j As Integer
Dim a2(1 to 3, 1 to 5) As Integer
WorkSheets("Лист1").Select
Cells.Clear
For i=1 To 3 'Формирование двумерного массива a2
      For j=1 To 5
            a2(I , j)=Int(Rnd(i*j)*100)
      Next j
Next i
For i = 1 To 3 'Вывод элементов массива a2(i , j) на рабочий лист
      For j = 1 To 5
          Cells(i+1 , j+1) = a2(I , j)
      Next j
Next i
End Sub
-----------------------------------------------------------------
Sub Dmas2()
Dim a(1 to 5) As Integer, k As Integer
Dim I As Integer, prom As Variant
WorkSheets("Лист1").Select
Cells.Clear
K=2
For i=1 To 5 'Ввод элементов массива a(i) с клавиатуры
    Do
        Prom=InputBox("Введите элемент a("Cint(i)")=")
        If Not IsNumeric(prom) Then MsgBox("Повторите ввод!")
    Loop Until IsNumeric(prom)
    a(i)=prom
Next i
For i=1 To 5 'Вывод элементов массива a(i) на рабочий лист
      Cells(k,2)="a("& Cint(i) & ")="
      Cells(k,3)=a(i)
      k=k+1
Next i
```

End Sub

### 6.1.2. Динамические массивы

Динамическим называется массив, размерность которого определяется в ходе выполнения программы. Синтаксис описания массива: Dim <Имя массива>() As <Тип>

Размерность массива устанавливается и изменяется с помощью инструкции ReDim <Имя массива>(<размерность>)

Для установки и изменения размерности массива без потери его содержимого применяется инструкция ReDim Preserve<Имя массива>(<размерность>). Такую инструкцию необходимо применять, например, при создании нового массива из существующего.

Пример:

```
Sub Mas4()
Dim a() As Integer
Dim I As Integer, k As Integer, j As Integer, N As Integer
Dim prom As Variant
WorkSheets("Лист1").Select
Cells.Clear
k=2
i=0
Do
    Prom=InputBox("Введите количество элементов N=)
    If Not IsNumeric(prom) Then MsgBox("Повторите ввод!")
Loop Until IsNumeric(prom)
N=prom
ReDim a(1 to N) As Integer 'Устанавливается фактическая размерность массива a
Do'Формирование массива a
    a(i)=Int(Rnd(i)*100)
    i=i+1
Loop Until i=N
For j=1 To N 'Вывод массива a[i] на рабочий лист
     Cells(k,j+1)=a(j)
```

```
Next j
End Sub
```

Array(<Список аргументов>)

С помощью такой инструкции создается массив типа Variant. Список аргументов представляет разделенный запятыми список значений, присваиваемых элементам массива.

Пример:

```
Dim День As Variant
День=Array("Пн", "Вт", "Ср", "Чт", "Пт", "Сб")
```

IsArray(<Имя переменной>)

Эта функция используется для проверки факта, является ли переменная типа Variant массивом. Она возвращает значение True, если переменная является массивом, и False в противном случае.

Erase(<Список массивов>)

С помощью этой инструкции повторно инициализируются элементы статических массивов, и освобождается память, отведенная для динамических массивов. Список представляет собой имена очищаемых массивов, разделенных запятой. В статических массивах их элементам вместо чисел присваиваются нулевые значения, а строки переменной длины становятся пустыми. В массивах типа Variant каждому элементу присваивается значение Empty.

## Технология выполнения лабораторной работы

1. Уяснить задачу и принять решение по ее выполнению.
2. Составить блок-схему решения задачи.
3. На основе блок-схемы составить код программы и ввести его в окно редактора VBA.
4. Протестировать программу и отладить ее при необходимости.
5. Результаты решения должны соответствовать поставленной задаче.

## Вопросы для защиты работы

1. Что такое массив?
2. Как объявляются статические массивы (одномерный и двумерный)? Приведите примеры.
3. Как объявляется динамический массив? Приведите примеры.
4. В каких случаях необходимо применять динамический массив?
5. Какой оператор нужно использовать после определения фактической размерности массива?
6. В каком случае необходимо применять оператор установки фактической размерности массива с сохранением его элементов?

## 6.2. Задание для лабораторной работы № 5. Одномерные массивы

(Время на выполнение 2 часа)

1. Сформировать массив **N** чисел, среди которых могут быть как положительные, так и отрицательные числа. Определить сумму и количество только отрицательных значений.
2. Сформировать массив **N** целых чисел. Определить сумму чисел, имеющих четные порядковые номера.
3. Сформировать массив **N** целых чисел. Подсчитать количество нулевых элементов и исключить их из массива.
4. Сформировать массив **N** целых чисел. Определить наличие среди них одинаковых. Нулевые значения не учитывать.
5. Сформировать массив **N** чисел, среди которых должны быть как положительные, так и отрицательные значения. Определить, сумма, каких чисел больше по абсолютной величине.
6. Сформировать массив **N** слов произвольной длины и найти самое длинное из них.
7. Сформировать массив **N** символов. Определить их коды и из них сформировать новый числовой массив
8. Сформировать массив **N** слов произвольной длины. Определить длину каждого из них и сформировать числовой массив.

9. Сформировать массив **N** целых чисел, удовлетворяющих условию: 32 =< a(i) <= 256. Считая эти числа кодами символов, определить эти символы и сформировать из них массив.
10. Сформировать массив **N** чисел, среди которых должны быть как положительные, так и отрицательные значения. Выбрать все положительные и все отрицательные числа и записать их в отдельные массивы.
11. Сформировать числовой массив заданной размерности. В полученных числах отделить целую часть, и записать в другой массив.
12. Из заданной строки текста сформировать массив символов, упорядочить его по возрастанию, преобразовать символы в коды и записать их в другой массив.

## 6.3. Задание для лабораторной работы № 6. Двумерные массивы

(Время на выполнение 2 часа)

1. Сформировать числовой массив из **M** строк и **N** столбцов. Подсчитать сумму значений элементов каждого столбца.
2. Создать строку из 25 букв русского алфавита. Используя функцию **Mid**, из букв этой строки сформировать массив, включающий 5 строк и 5 столбцов.
3. Создать строку из 25 букв русского алфавита. Определить код каждой буквы и сформировать числовой массив из 5 строк и 5 столбцов
4. Сформировать числовой массив из **M** строк и **N** столбцов. Подсчитать сумму значений элементов каждой строки.
5. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Подсчитать количество и сумму всех чётных чисел массива.
6. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Подсчитать количество и сумму всех нечётных чисел массива.
7. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Подсчитать сумму чисел в чётных строках массива.

8. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Подсчитать сумму чисел в нечётных строках массива.
9. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Подсчитать сумму чисел в чётных столбцах массива.
10. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Определить номер строки, сумма чисел которой наибольшая.
11. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Определить номер столбца, в котором число нечётных элементов больше числа чётных элементов.
12. Сформировать массив натуральных чисел из **M** строк и **N** столбцов. Выбрать из массива все числа, которые делятся на 3 без остатка.

## 7. Лабораторная работа № 7

### 7.1. Задание. По знаковой модели алгоритма составить блок-схему и программу

(Время на выполнение 4 часа)

### Вопросы для защиты работы

1. Что такое алгоритм?
2. Свойства алгоритма.
3. Способы записи алгоритма.
4. Основные элементы блок-схемы.
5. Виды алгоритмов.
6. Отличительные особенности алгоритмов циклов с предусловием и постусловием.

1. Варианты 1-5. Вычислить значение переменной z для исходных данных: x, y, z.
2. Варианты 6-13. Вычислить значение переменной S для значений N элементов одномерного массива a[i].

1) x=12, y=3, z=10. Алгоритм А1.
2) x=14, y=5, z=12. Алгоритм А1.

3) x=10, y=2, z=8. Алгоритм А1.
4) x=6, y=15, z=11. Алгоритм А2.
5) x=4, y=13, z=9. Алгоритм А2.
6) n=10, a[i] = 35; 22; 5; 9; 28; 14; 2; -18; 33; 24. Алгоритм А3.
7) n=10, a[i] =-24; -7; 9; 14; -17; 21; 27; -15; 19; -28. Алгоритм А3.
8) n=10, a[i] = 35; 22; 5; 9; 28; 14; 2; -18; 33; 24. Алгоритм А4.
9) n=10, a[i] =-27;-4; 20; -15; -16; -13; -9; -11; 20; -12. Алгоритм А4.
10) n=10, a[i] = 35; -22; 5; 9; 28; 14; -2; -18; 33; 24. Алгоритм А5.
11) n=10, a[i] =-27; -4; 20; -15; -16; -13; -9; -11; 20; -12. Алгоритм А5.
12) n=10, a[i] =-27; -4; 20; -15; -16; -13; -9; -11; 20; -12. Алгоритм А6.
13) n=10, a[i] = 35; 22; 5; -9; 28; 14; 2; -18; -33; 24. Алгоритм А6.

**Алгоритм А1**

```
нач
цел x, y, z
x = x – z
y = y – x
z = z + 2
если z <= x
    то x = x – z
       y = y – x
       z = x + y + 1
иначе z = z – x
   если z <= y
      то x = x + z
         y = y + x
         z = x + y – 1
иначе x = x – z
      y = y + x
      z = x + y + 1
      все
все
x = x + 2
```

```
   y = y + 3
   z = z – 2
   конец
```

**Алгоритм А2**

```
нач
цел x, y, z
x = x + 7
y = y – 2
z = z + 2
если  x <= z
то если y < x
     то  y = 5*x
     иначе x = 2*y
     все
иначе если x > y
          то  y = 4*x
          иначе x = 3* y
          все
все
x = x + y
y = y + z
z = x + y + z
кон
```

**Алгоритм А3**

```
нач
цел a[10], n, i
вещ s
ввод n
i = 1
нц
пока i <= n
     ввод a[i]
     i = i+ 1
кц
```

```
s = 0
нц
для i от 1 до n
      если (a[i]>=10) и (a[i]<=99)
            s=s + a[i]
      все
кц
s=s/2
вывод s
кон
```

**Алгоритм А4**

```
нач
цел a[10], n, I, j
вещ s
ввод n
нц
для i от 1 до n
      ввод a[i]
кц
i = 1
j = 0
s = 0
нц
   если a[i]>0
        то  s=s + a[i]
        j=j+1
   все
   i = i+1
до i>n
кц
s=s/j
вывод s
кон
```

**Алгоритм А5**

```
нач
цел a[10], n, i, s, m1, m2
ввод n
нц
пока i<=n
      ввод a[i]
      i=i+1
кц
m1=a[1]
m2=a[1]
нц
для i от 2 до n
        если a[i]<m1
            то  m1=a[i]
        все
        если a[i]>m2
            то  m2=a[i]
        все
кц
s=m2-m1
вывод s
кон
```

**Алгоритм А6**

```
нач
цел a[10], n, I, s
ввод n
i=1
нц
      ввод a[i]
      i=i+1
до i>n
кц
i=2
s=0
```

```
нц
пока i<=n
      если (a[i]<0) и (a[i]>-15)
            то s=s + a[i]
      все
i=i+2
кц
s=2*s
вывод s
конец
```

По теоретическому курсу проводится тестирование.

## Литература


1. Таганов Л. С., Пимонов А. Г. Информатика: [Электронный ресурс]: учеб. пособие для студентов техн. направлений и специальностей вузов / Л. С. Таганов, А. Г. Пимонов; ГУ КузГТУ. – Кемерово, 2010. – 330 с. http://library.kuzstu.ru/meto.php?n=90457&.type=utchposob:common
2. Информатика. Базовый курс : учеб. пособие для студентов втузов / под ред. С. В. Симоновича. - 3-е изд. - СПб.: Питер, 2012. - 640 с.
3. Таганов, Л. С. Конспекты лекций по курсу «ИНФОРМАТИКА»: [Электронный ресурс]: для студентов всех форм обучения специальности 130400.65 «Горное дело» / Л. С. Таганов; КузГТУ. – Кемерово, 2013. – 290 с. http://library.kuzstu.ru/meto.php?n=91010&.type=utchposob:common