

**РОСТОВСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ
УНИВЕРСИТЕТ “РИНХ”**

**Факультет информатизации и управления
Кафедра Информационных технологий**

Е.В. Ефимова

**ИНФОРМАТИКА
VBA в Office**

**Ростов-на-Дону
2009**

Печатается по решению кафедры Информационных технологий

Автор: Ефимова Е.В., кандидат экономических наук, доцент

Рецензенты:

Кандидат физико-математических наук, доцент Иванов В.В.

Кандидат экономических наук, доцент Фоменко Н.М.

УМП в доступной форме рассматривает язык объектно-ориентированного программирования Visual Basic Application и его использование для разработки офисных приложений в пакетах MS Office: Excel, Word. Благодаря этим пакетам прикладных программ можно создавать интегрированные документы, опирающиеся на данные различных приложений.

УМП ориентировано на студентов экономических вузов, и широкого круга пользователей, работающих в среде Windows MS Office.

**© Ростовский государственный
Экономический университет «РИНХ», 2009**

Оглавление

Введение в VBA для приложений MS OFFICE	4
Лабораторная работа №1. Пользовательские функции VBA.	21
Лабораторная работа №2. Разработка функций для реализации линейных и разветвляющихся алгоритмов.	22
Лабораторная работа №3. Расчет комиссионных.	24
Лабораторная работа №4. Построение диаграммы для функции.	26
Лабораторная работа №5. Элементы управления.	27
Лабораторная работа №6. Создание кнопочного сценария расчета.	28
Лабораторная работа №7. Автоматизация расчета эффективности капиталовложений с помощью счетчика.	30
Лабораторная работа №8. Создание пользовательской формы.	32
Лабораторная работа №9. Создание пользовательской формы. Пример использования рамки и переключателя.	34
Лабораторная работа №10. Создание пользовательской формы. Пример использования флажков.	36
Лабораторная работа №11. Пример создания диалогового окна	39
Лабораторная работа №12. Табличная база данных туристической фирмы.	40
Лабораторная работа №13. Самостоятельная работа.	48
Лабораторная работа №14. Работа с элементами управления	50
Лабораторная работа №15 Самостоятельная работа.	54
Список литературы	56

Введение в VBA для приложений MS OFFICE

Подобно другим языкам программирования VBA позволяет создавать полностью автоматические программные продукты. Однако, прежде всего, VBA – это инструмент разработки приложений MS Office. Дело в том, что VBA является общей языковой платформой для всех приложений MS Office. Поэтому он встроен во все приложения: Excel, Word, Access и др. Это с одной стороны значительно расширяет функциональные возможности каждого приложения, а с другой позволяет объединять данные из нескольких приложений в одном документе. Например, можно открыть базу данных Access, проанализировать ее данные с помощью встроенных или созданных средств в Excel, а результаты анализа вывести в документ Word.

Язык VBA является производным от языка *Visual Basic (VB)*. Синтаксис этих языков практически одинаков. Основное различие их заключается в том, что VB имеет собственную среду разработки, а VBA использует среду, встроенную в приложение MS Office – редактор VBA. В силу этого, с помощью VB можно создать полностью самостоятельный программный продукт, в то время как проекты VBA могут быть выполнены только с помощью приложения, которое поддерживает VBA.

Структура программ на VBA

Программы на VBA хранятся в проектах. Проект содержит модули различных типов, а модули включают различные процедуры. Проект может содержать несколько модулей. Имеются следующие типы модулей: стандартные модули - это модули, в которых можно описать доступные во всем проекте процедуры; модули класса содержат описание объекта, который является членом класса. Процедуры, написанные в модуле класса, используются только в этом модуле.

Среди модулей класса выделяют модули форм и отчетов, которые связаны с конкретной формой или отчетом. Модули форм и отчетов часто содержат процедуры обработки событий, которые срабатывают в ответ на событие в форме или отчете. Процедуры обработки событий используются для управления поведением форм и отчетов и их реакцией на действия пользователя типа щелчка мыши на кнопке.

Модули содержат описания и процедуры - наборы описаний и инструкций, сгруппированных для выполнения. Существует три типа процедур:

- процедура Sub - набор команд, с помощью которого можно решить определенную задачу. При ее запуске выполняются команды процедуры, а затем управление передается в приложение пакета MS Office или процедуру, которая вызвала данную процедуру.
- процедура Function (функция) также представляет собой набор команд, который решает определенную задачу. Различие заключается в том, что такие процедуры обязательно возвращают значение, тип которого можно описать при создании функции.

○процедура Property используется для ссылки на свойство объекта. Данный тип процедур применяется для установки или получения значения пользовательских свойств форм и модулей.

Для создания модуля в любом приложении MS Office необходимо выбрать команду меню Сервис – Макрос - Редактор Visual Basic. В окне "Проект" необходимо щелкнуть правой кнопкой мыши на любом элементе либо в окне редактора выбрать команду меню Вставка, а далее тип модуля. При выборе формы (Userform) для перехода к ее модулю используется команда Вид - Программа или кнопка "Программа" в окне "Проект".

Самое начало модуля называется общей областью, в которой располагаются общие описания, например, типа данных, используемого по умолчанию (DefТип), инструкция Option Explicit, требующая явного описания всех используемых в модуле переменных, а также описания общих (глобальных) для всех модулей и для данного модуля переменных.

Операторы описания

Объявление переменной производится одним из операторов Dim, Static, Private, Public, за которым следует имя переменной и необязательная часть с ключевым словом As, после которого задается тип переменной, например Dim name [As type]. Оператор Public используется только вне модуля, в его общей части и делает описываемую переменную доступной из всех процедур всех модулей проекта. Оператор Private служит для объявления переменной уровня модуля, доступной только процедурам данного модуля. Можно использовать также оператор Dim, но применение Private предпочтительнее как противоположное Public.

Переменные могут быть объявлены внутри процедуры операторами Dim или Static. Такие переменные называют также локальными, поскольку доступны только в той процедуре, в которой они объявлены. Данное свойство (область видимости) позволяет использовать одинаковые имена переменных в разных процедурах, не опасаясь конфликтов или случайных изменений значений переменных. Время жизни локальных переменных, объявленных с помощью оператора Dim равно времени работы процедуры и по ее окончании значения таких переменных теряются.

Переменные, объявленные с помощью оператора Static сохраняют свои значения в течении всего времени выполнения приложения. При повторном входе в процедуру, где описана такая переменная, ее значение сохраняется.

Операторы Public и Private можно применять при описании констант и процедур, что позволяет указать их область видимости. Для процедур возможно также применение оператора Static, что позволяет сделать все переменные в процедуре статическими:

Static Function Total (num) as Integer

Это приводит к тому, что все локальные переменные в процедуре становятся статическими, независимо от того, как они определены; операторами Static, Dim, Private или неявным образом.

Операторы присваивания

Инструкция Let Присваивает значение выражения переменной или свойству:

[Let] имяПеременной = выражение

Явное использование ключевого слова Let зависит от вкуса пользователя, обычно это слово опускают.

Значение выражения может быть присвоено переменной, только если оно имеет совместимый с этой переменной тип данных. Невозможно присвоить строковое выражение числовой переменной или числовое выражение строковой переменной. Такая попытка приведет к ошибке во время компиляции.

Переменным типа Variant могут присваиваться как строковые, так и числовые выражения. Однако обратное не всегда верно. Любое значение типа Variant, за исключением значения Null, допускает присвоение строковой переменной, но только значение типа Variant, которое может рассматриваться как число, может быть присвоено числовой переменной. Пользуйтесь функцией IsNumeric для определения возможности преобразования значения Variant в числовое значение.

Внимание! Присвоение выражения с одним из числовых типов переменной с другим числовым типом данных преобразует значение выражения в тип данных результирующей переменной.

Инструкция Let может быть использована для присвоения одной переменной-записи другой, только если обе переменные имеют одинаковый определяемый пользователем тип. Для присвоения переменных-записей различных определяемых пользователем типов используется инструкция LSet. Для присвоения переменным ссылок на объекты применяется инструкция Set.

Управляющие структуры позволяют управлять последовательностью выполнения программы. Без операторов управления все операторы программы будут выполняться слева направо и сверху вниз. Однако иногда требуется многократно выполнять некоторый набор инструкций автоматически, либо решить задачу по-другому в зависимости от значения переменных или параметров, заданных пользователем во время выполнения. Для этого служат конструкции управления и циклы.

VBA поддерживает следующие конструкции принятия решений:

```
If . . . Then  
If . . . Then . . . Else  
Select Case
```

Конструкция If . . . Then

Конструкция If . . . Then применяется, когда необходимо выполнить один или группу операторов в зависимости от некоторого условия.

Синтаксис этой конструкции позволяет задавать ее в одной строке или в нескольких строках программы:

```
If условие Then выражение
```

```
If условие Then
выражение
End If
```

Обычно условие является простым сравнением, но оно может быть любым выражением с вычисляемым значением. Это значение интерпретируется как False (Ложь), если оно нулевое, а любое ненулевое рассматривается как True (Истина). Если условие истинно, то выполняются все выражения, стоящие после ключевого слова Then. Для условного выполнения одного оператора можно использовать как синтаксис для одной строки, так и синтаксис для нескольких строк (блоковую конструкцию).

Следующие два оператора эквивалентны:

```
If anyDate < Now Then anyDate = Now
If anyDate < Now Then
anyDate = Now
End If
```

Заметим, что синтаксис оператора If . . . Then для одной строки не использует оператор End If. Чтобы выполнить последовательность операторов, если условие истинно, следует использовать блоковую конструкцию If . . . Then . . . End If.

```
If anyDate < Now Then
anyDate = Now
Timer.Enabled = False ' Запретить таймер.
End If
```

Если условие ложно, то операторы после ключевого слова Then не выполняются, а управление передается на следующую строку (или строку после оператора End If в блочной конструкции).

Конструкция If . . . Then . . . Else

Определяет несколько блоков операторов, один из которых будет выполняться в зависимости от условия:

```
If условие1 Then
выражение1
ElseIf условие2 Then
выражение2
. . .
Else
выражение-n
End If
```

При выполнении сначала проверяется условие1. Если оно ложно, VBA проверяет следующее условие2 и т. д., пока не найдет истинного условия. Найдя его, VBA выполняет соответствующий блок операторов и затем передает управление инструкции, следующей за оператором End if. В данную конструкцию можно включить блок оператора Else, который VBA выполняет, если не выполнено ни одно из условий.

Конструкция If . . . Then . . . ElseIf в действительности всего лишь специальный случай конструкции If . . . Then . . . Else. Заметим, что в данной конструкции может быть любое число блоков ElseIf, или даже ни одного. Блок Else можно включать независимо от присутствия или, наоборот, отсутствия блоков ElseIf.

Заметим, что можно добавить любое число блоков ElseIf в конструкцию If . . . Then. Однако количество блоков ElseIf может стать настолько большим, что конструкция If . . . Then станет очень громоздкой и неудобной. В подобной ситуации следует применять другую конструкцию принятия решения - Select Case.

Конструкция Select Case

Конструкция Select Case является альтернативой конструкции If . . . Then . . . Else в случае выполнения блока, состоящего из большого набора операторов. Конструкция Select Case предоставляет возможность, похожую на возможность конструкции If . . . Then . . . Else, но в отличие от нее она делает код более читаемым при наличии нескольких вариантов выбора. Конструкция Select Case работает с единственным проверяемым выражением, которое вычисляется один раз при входе в эту конструкцию. Затем VBA сравнивает полученный результат со значениями, задаваемыми в операторах Case конструкции. Если найдено совпадение, выполняется блок операторов, ассоциированный с оператором Case:

```
Select Case проверяемое_выражение
[Case список_выражений1
[блок_операторов1]]
[Case список_выражений2
[блок_операторов2]]
. . .
[Case Else
[блок_операторовn]]
End Select
```

Каждый список выражений является списком из одного или более значений. Если в одном списке больше одного значения, они отделяются запятыми. Каждый блок операторов содержит несколько операторов или ни одного. Если окажется, что вычисленному значению проверяемого выражения соответствуют значения из нескольких операторов Case, то выполняется блок операторов, ассоциированный с первым оператором Case из всех найденных соответствий. VBA выполняет блок операторов, ассоциированный с оператором Case Else (заметим, что он необязателен), если не найдено ни одного соответствия проверяемого значения выражения и значений из всех списков операторов Case.

Заметим, что конструкция Select Case вычисляет выражение только один раз при входе в нее, а в конструкции If . . . Then . . . Else вычисляются различные выражения для каждого оператора ElseIf. Конструкцию If . . . Then

. . . Else можно заменить конструкцией Select Case, только если оператор If и каждый оператор Elseif вычисляют одно и то же выражение.

Операторы цикла.

Циклы позволяют выполнить одну или несколько строк кода несколько раз. VBA поддерживает следующие циклы:

For...Next

For Each...Next

Do... Loop

Конструкция For . . . Next. Когда число повторений известно заранее, используют цикл For . . . Next. В цикле For используется переменная, называемая переменной цикла или счетчиком цикла, которая увеличивается или уменьшается на заданную величину при каждом повторении цикла.

Синтаксис этой конструкции следующий:

```
For counter = start To end [Step increment]
```

```
операторы
```

```
Next [counter]
```

Параметры counter (счетчик), start (начало цикла), end (конец цикла) и increment (приращение) являются числовыми.

Примечание. Параметр increment может быть как положительным, так и отрицательным. Если он положителен, параметр start должен быть меньше или равен параметру end, иначе цикл не будет выполняться. Если параметр increment отрицателен, то параметр start должен быть больше или равен значению параметра end, чтобы выполнялось тело цикла. Если параметр Step не задан, то значение параметра increment по умолчанию равно 1.

VBA выполняет цикл For в следующей последовательности:

1. Устанавливает значение переменной цикла counter в значение start.
2. Сравнивает значение переменной цикла counter и значение параметра end. Если переменная counter больше, VBA завершает выполнение цикла. (Если значение параметра increment отрицательно, то VBA прекращает выполнение цикла при условии, что значение переменной цикла counter меньше значения параметра end.)
3. Выполняет операторы тела цикла statements.
4. Увеличивает значение переменной цикла counter на 1 или на величину значения параметра increment, если он задан.
5. Повторяет шаги со 2 по 4.

Конструкция For Each . . . Next

Цикл For Each . . . Next похож на цикл For . . . Next, но он повторяет группу операторов для каждого элемента из набора объектов или из массива, вместо повторения операторов заданное число раз. Он особенно полезен, когда неизвестно, сколько элементов содержится в наборе.

Синтаксис конструкции цикла For Each . . . Next таков:

```
For Each element In group
```

```
операторы
```

```
Next element
```

Следует помнить следующие ограничения при использовании цикла For Each . . . Next:

- Для наборов параметр element может быть только переменной типа variant, общей переменной типа object или объектом, перечисленным в Object Browser
- Для массивов параметр element может быть только переменной типа Variant
- Нельзя использовать цикл For Each . . . Next с массивом, имеющим определенный пользователем тип, так как переменная типа variant не может содержать значение определенного пользователем типа

Конструкция Do...Loop

Цикл Do применяется для выполнения блока операторов неограниченное число раз. Существует несколько разновидностей конструкции Do . . . Loop, но каждая из них вычисляет выражение-условие, чтобы определить момент выхода из цикла. Как и в случае конструкции If . . . Then условие должно быть величиной или выражением, принимающими значение False (нуль) или True (не нуль).

В следующей конструкции Do . . . Loop операторы выполняются до тех пор, пока значением условия является True (Истина):

```
Do While условие
    операторы
Loop
```

Выполняя этот цикл, VBA сначала проверяет условие. Если условие ложно (False), он пропускает все операторы цикла. Если оно истинно (True), VBA выполняет операторы цикла, снова возвращается к оператору Do While и снова проверяет условие.

Следовательно, цикл, представленный данной конструкцией, может выполняться любое число раз, пока значением условия является не нуль или True (Истина). Отметим, что операторы тела цикла не выполняются ни разу, если при первой проверке условия оно оказывается ложным (False).

Другая разновидность конструкции Do . . . Loop сначала выполняет операторы тела цикла, а затем проверяет условие после каждого выполнения. Эта разновидность гарантирует, что операторы тела цикла выполняются по крайней мере один раз:

```
Do
    операторы
Loop
While условие
```

Две другие разновидности конструкции цикла аналогичны предыдущим, за исключением того, что цикл выполняется, пока условие ложно (False):

- Цикл не выполняется вообще или выполняется много раз:

```
Do Until условие
    операторы Loop
```

- Цикл выполняется по крайней мере один раз:

```
Do
```

операторы
Loop Until условие

Вложенные циклы.

Можно помещать структуры управления внутрь других структур управления (например, блок If . . . Then внутрь цикла For . . . Next). Говорят, что структура управления, помещенная внутрь другой структуры управления, является вложенной.

Глубина вложения управляющих структур в VBA не ограничена. Для улучшения читаемости кода принята практика смещения тела конструкции принятия решения или цикла в программе в случае использования вложенных структур управления.

При вложении в цикл одного или несколько других циклов говорят о вложенных циклах, в которых различают внешние (охватывающие) и внутренние (вложенные) циклы.

Заметим, что первый оператор Next закрывает внутренний цикл For, а последний оператор Next закрывает внешний цикл For. Точно так же и для вложенных операторов If, операторы End If автоматически применяются для закрытия ближайшего к нему оператора If. Вложенные структуры Do . . . Loop работают подобным же образом: самый дальний оператор Loop соответствует самому дальнему оператору Do.

При вводе/выводе элементов двумерного массива на рабочий лист Microsoft Excel удобно применять пользовательские процедуры ввода/вывода:

```
Sub readcell(i As Integer, j As Integer, val As Variant)
    val = Лист1.Cells(i, j).Value
End Sub
Sub outcell(i As Integer, j As Integer, val As Variant)
    Лист1.Cells(i, j).Value = val
End Sub
```

где I - номер строки, j - номер столбца рабочего листа.

Выход из структур управления

Оператор Exit позволяет выходить непосредственно из цикла For, цикла Do, процедуры Sub или процедуры Function. Синтаксис оператора Exit прост:

```
For counter = start To end [Step -increment]
    [блок операторов]
[Exit For]
[блок операторов]
Next [counter]
Do [(While | Until} условие]
    [блок операторов]
[Exit Do]
[блок операторов]
Loop
```

Exit For внутри цикла For и Exit Do внутри цикла Do могут появиться сколько угодно раз.

Оператор Exit Do работает со всеми разновидностями синтаксиса цикла Do.

Операторы Exit For и Exit Do применяются, если необходимо завершить цикл немедленно, не продолжая дальнейших итераций или не ожидая выполнения блока операторов в теле цикла.

При использовании оператора Exit для выхода из цикла значения переменной цикла зависят от того, каким образом завершается выполнение цикла:

- При нормальном завершении цикла значение переменной цикла имеет на единицу больше верхней границы числа циклов
- При преждевременном завершении цикла переменная цикла сохраняет свое значение, которое она получила с учетом обычных правил
- При завершении цикла по концу набора переменная цикла имеет значение Nothing (Ничего), если она является переменной типа object (Объект), или значение Empty (Пусто), если она является переменной типа Variant

Перечень основных элементов управления

Рассмотрим набор элементов управления. Помимо элементов управления, появляющихся на стандартной панели, щелчок по ней правой кнопки открывает большой список дополнительных элементов управления, более того, пользователь имеет возможность расширить этот список. Мы ограничимся сейчас рассмотрением некоторого стандартного набора, который включает следующие объекты:

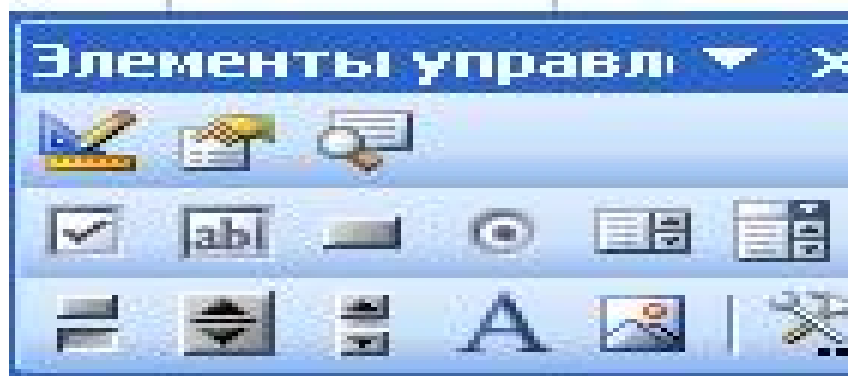


Рис.1. Панель инструментов Элементы управления.

- CheckBox — флажок;
- ComboBox — комбинированный список (поле со списком);
- CommandButton — командная кнопка;
- Image — изображение (окно изображения);
- Label — метка (надпись, статический текст);
- ListBox — список (окно списка);
- OptionButton — переключатель (кнопка зависимого выбора);
- ScrollBar — полоса прокрутки;
- SpinButton — счетчик (ворот);

- **TextBox** — поле ввода (окно редактирования, текстовое поле);
- **ToggleButton** — выключатель.

А кроме них в верхней части панели имеются кнопки **Режим Конструктора**, **Свойства** и **Исходный текст**. С их помощью можно вносить соответствующие изменения в описание и состояние объекта.

CheckBox — флажок (кнопка выбора)

Флажок, или кнопка независимого выбора, позволяет пользователю указать, выбирает или не выбирает он опцию (обычно ее название является также названием флажка, т. е. значением свойства `Caption`). В зависимости от значения свойства `TriState` у флажка может быть два или три состояния. По умолчанию значение этого свойства `False`, что соответствует двум значениям состояния флажка: `True` — флажок выбран, включен и `False` — флажок не выбран, выключен. Если для свойства `TriState` установить `True`, у флажка появится также нейтральное состояние `Null`. Состояния флажка передаются в программу через свойство `Value` (Значение).

ComboBox — комбинированный список

Комбинированный список — это элемент управления, соединяющий поле ввода с кнопкой и раскрывающимся списком. Работая с ним, пользователь может ввести значение непосредственно в поле ввода либо выбрать одно из значений в списке. Список состоит из строк данных. Данные в строке могут располагаться в одном или нескольких столбцах. Рассмотрим основные свойства объекта `ComboBox`.

- **ListCount** определяет, сколько элементов находится в списке. Свойство доступно только для чтения и изменяется автоматически вместе с добавлением (удалением) элементов в список.
- **ListRows** задает число одновременно видимых строк списка.
- **ListIndex** задает номер выбранной строки; возможные значения от -1 до `ListCount-1`, т. е. значение `ListCount` всегда на 1 больше максимального значения свойства `ListIndex`, так как нумерация строк начинается с 0. Если ни один элемент в списке не выбран, `ListCount` равно 0, а значением `ListIndex` будет -1.
- **ColumnCount** определяет число столбцов в выводимом на экран списке; если оно равно 0, столбцы не выводятся, при значении -1 выводятся все столбцы.
- **ColumnWidths** задает ширину каждого столбца для списков с несколькими столбцами. Значение этого свойства — строка, в которой размеры столбцов перечислены через точку с запятой. Пустое значение или -1 означают, что ширина столбца вычисляется автоматически, 0 — отсутствие столбца, значения > 0 задают ширину столбца в точках. Можно также рядом с числовым значением указывать другую единицу измерения. По умолчанию ширина столбца

не меньше 1 дюйма (72 точек).

- **TextColumn** задает номер столбца, видимого пользователю. Нумерация столбцов начинается с 1. 0 означает, что будет виден только выбранный элемент из строки `ListIndex`. При значении 1 выводится первый столбец, чья ширина, установленная свойством `ColumnWidths`, больше 0.
- **BoundColumn** указывает столбец со значением данных (свойством `Value`) в списке со многими столбцами. Если свойство равно 0, значением свойства `Value` будет номер `ListIndex` выбранной пользователем на экране строки. Если `BoundColumn` > 0, значение свойства `Value` берется из указанного столбца. Таким образом, пользователь может выбрать на экране один элемент, а в качестве значения этого выбора в программе можно задать другой.
- Задать элементы списка можно программно, используя метод `AddItem` (например, в процедуре инициализации диалогового окна), либо установив при проектировании свойство **RowSource**. Его значение — строка, задающая диапазон ячеек Excel, из которых будут браться элементы списка.
- **List** — двумерный массив с элементами списка. Обращение к нему: *объект.List(строка, столбец)*
а элементы этого массива имеют тип `Variant`. Нумерация строк и столбцов начинается с 0. Это свойство можно использовать для инициализации списка с несколькими столбцами в элементах управления `ComboBox` и `ListBox`.
- **ListStyle** определяет внешний вид списка. Если оно равно `fmListStylePlain = 0`, элементы списка выводятся в обычном виде без кнопок слева. Если же его значение — `fmListStyleOption = 1`, слева от каждого элемента списка выводится кнопка. Для списков с единственным выбором это кнопка-переключатель (`OptionButton`), а для списков с множественным выбором — кнопка-флажок (`CheckBox`). При выборе элемента в соответствующей кнопке появляется метка.
- **MatchRequired** и **MatchEntry** определяют поведение комбинированного списка при вводе пользователем данных в поле ввода. Если `MatchRequired` равно `True`, вводимый пользователем текст может стать значением элемента, лишь когда он совпадает с одним из элементов списка. По умолчанию ему присвоено `False`, что не требует от вводимых пользователем данных совпадения с элементами списка. Свойству `MatchEntry` по умолчанию устанавливается 1, означающее, что при вводе пользователем очередного символа в списке ищется первый элемент, для которого введенное слово является префиксом и выводится в качестве значения. Если `MatchEntry` равно 0, поиск происходит по первой букве слова, т. е. при повторном выборе одной

и той же буквы в качестве значений перебираются все элементы списка, начинающиеся на нее. При `MatchEntry` равном 2 список не реагирует на набираемый пользователем текст. Этот текст доступен через свойство `Text`.

CommandButton — командная кнопка

Командная кнопка запускает на выполнение действия системы, которые обычно задаются в процедуре обработки события `Click` (щелчок кнопки). Вот ее некоторые свойства.

- Название кнопки — значение свойства **Caption** — можно установить при проектировании диалогового окна, а затем менять из программы.
- Булево свойство **Cancel** определяет, является ли данная кнопка кнопкой отказа. Если свойство равно `True`, то нажать эту кнопку можно, щелкнув по ней мышью, либо нажав клавишу `<Esc>`, либо нажав клавишу `<Enter>` в тот момент, когда она находится в фокусе. По умолчанию задается `False`. Это свойство может быть установлено только для одной кнопки в окне (это отслеживается системой автоматически). Типичные действия, которые должны выполняться при выборе кнопки отказа, состоят в восстановлении состояний и значений элементов управления и связанных с ними данных, измененных пользователем во время работы в диалоговом окне. Их следует описать в процедуре обработки события `Click` для данной кнопки.
- Булево свойство **Default** определяет, выбирается ли кнопка по умолчанию, если пользователь щелкает окно или нажимает клавишу `Enter`, когда фокус не находится на другой командной кнопке. По умолчанию свойству задается значение `False`.
- Булево свойство **TakeFocusOnClick** определяет, переместится ли фокус на данную кнопку после выбора ее щелчком. Иногда для этого свойства удобно установить `False`, чтобы, щелкнув кнопку, выполнить действие над содержимым элемента, находящегося в фокусе.

Frame — рамка (группы)

Элемент управления `Frame` служит для явно видимого объединения в группу нескольких других элементов управления. Кнопки-переключатели (`OptionButton`), помещенные в рамку, автоматически являются взаимоисключающими, т. е. при выборе одной из них остальные сразу же отключаются (получают значение `False`). Поведение других элементов, помещенных в рамку, не меняется.

Заголовок на верхней границе рамки задается свойством `Caption`. Как и для диалогового окна, для рамки можно задать рисунок, являющийся фоном (свойство `Picture`), определить область прокрутки и вид полосы прокрутки (свойства `ScrollLeft`, `ScrollTop`, `ScrollHeight`, `ScrollWidth`, `ScrollBars`).

Свойство `Zoom` определяет коэффициент уменьшения или увеличения

изображений всех элементов внутри рамки (измеряется в процентах и принимает значения от 10% до 400%).

Image — изображение

Элемент Image — это прямоугольное графическое изображение в диалоговом окне. Его используют как для улучшения внешнего вида окна, так и для показа пользователю информации, представляемой в виде фотографий, рисунков, графиков, диаграмм. Рисунок в элементе Image можно масштабировать, изменять выводимую часть (если рисунок больше изображения на экране), можно динамически изменять размеры изображения, но рисунок нельзя редактировать в диалоговом окне.

Какой именно рисунок или графический образ показывается на экране, определяет свойство **Picture**. Его значением может быть изображение в одном из графических форматов: .bmp, .cur, .gif, .ico, .jpg и .wmf. Задать значение этого свойства можно при проектировании окна из списка свойств элемента Image. Для этого, щелкнув кнопку (...) в строке со свойством Picture, вызовите программу поиска и загрузки графических файлов, найдите файл с нужным изображением и выберите кнопку Open. Чтобы вывести или изменить изображение программно, используется функция LoadPicture.

Label — метка (надпись, статический текст)

Один из самых простых элементов управления Label (метка) служит для вывода разного рода надписей в диалоговом окне. Часто эти надписи именуют или объясняют другие элементы окна. При работе программы текст метки может меняться. Основное свойство метки — **Caption** — содержит в качестве значения текст метки.

ListBox — список

Элемент управления ListBox выводит на экран окно со списком значений, позволяя пользователю выбрать из них одно или более. В варианте с выбором одного элемента ListBox ведет себя, как и описанный выше список в элементе ComboBox. Значение передается в свойстве Value, такую же роль играют свойства ColumnCount, ColumnWidth, ControlSource, RowSource, BoundColumn, List-Index, TextColumn и др. Главное отличие — способность обеспечить множественный выбор. Режим выбора определяется свойством **MultiSelect**, по умолчанию его значение fmMultiSelectSingle = 0, что соответствует выбору одного значения в списке. Значение fmMultiSelectMulti = 1 задает режим множественного выбора, при котором выбор и отмена выбора элемента осуществляются щелчком мышью или нажатием клавиши «пробел». Значение fmMulti-SelectExtended = 2 задает режим, при котором выбор осуществляется щелчком или нажатием клавиши Shift, а расширить область выделенных элементов можно, используя клавиши-стрелки при нажатой клавише Shift.

OptionButton — кнопка-переключатель

Кнопка-переключатель (радио – кнопка или просто переключатель), как

и флажок, показывает, выбран ли элемент. Отличие в том, что из нескольких переключателей, объединенных в группу, выбран может быть только один. Объединять переключатели в группу можно двумя способами. Первый: сначала создается элемент `Frame` — рамка группы, затем в этой рамке размещают переключатели, образующие группу. Они автоматически становятся взаимоисключающими. Другой способ — присвоив группе имя, установить его как значение свойства **GroupName** для всех переключателей этой группы. У каждого способа свои плюсы и минусы. В первом случае рамка вокруг переключателей помогает пользователю зрительно идентифицировать группу, но она занимает дополнительную площадь в окне (а ведь может потребоваться разместить несколько групп переключателей!), да и рамка перекрывает фон окна. Второй вариант гибче: кнопки одной группы могут размещаться в окне произвольным способом, но сложнее обеспечить ясное выделение каждой группы кнопок.

Перемещение по кнопкам одной группы осуществляется с помощью клавиш-стрелок, при этом фокус перемещается только по элементам данной группы, даже если при обходе с помощью клавиши `Tab` между кнопками группы включены другие элементы управления. Выбор переключателя, находящегося в фокусе, происходит при нажатии клавиши «пробел» или при щелчке любого переключателя группы — его свойство `Value` получает значение `True`, а для всех остальных переключателей группы устанавливается значение `False`.

ScrollBar — полоса прокрутки

Элемент управления `ScrollBar` представляет вертикальную или горизонтальную полосу, на краях которой расположены кнопки прокрутки, а внутри перемещается бегунок. Значение `Value`, устанавливаемое в полосе прокрутки или возвращаемое ей, — число, определяемое положением бегунка и границами, определенными в свойствах **Min** и **Max**. Рекомендуемые значения этих границ от `-32767` до `+32767` (по умолчанию установлен диапазон `[0, 32767]`). Если отношение длин левого и правого отрезков, на которые полоса прокрутки делится бегунком, — **L:R**, то

Value = (Min * R + Max * L) / (L + R).

Обычно полоса прокрутки используется в паре с другим элементом управления, в котором может отображать или с которого может получать свое значение..

И еще несколько важных свойств полосы прокрутки.

- Горизонтальная или вертикальная ориентация полосы прокрутки определяется свойством **Orientation**. При его значении по умолчанию `fmOrientationAuto = 1`, ориентация полосы определяется автоматически в зависимости от ее размера по горизонтали и вертикали (большой размер задает ориентацию); `FmOrientationVertical = 0` задает вертикальную ориентацию полосы, `FmOrientationHorizontal = 1` — горизонтальную.

- Свойства **LargeChange** и **SmallChange** определяют, на сколько изменится значение Value при одном щелчке поверхности полосы между кнопкой прокрутки и бегунком в первом случае, и при щелчке кнопки прокрутки — во втором. Эти же свойства указывают, насколько при этом смещается бегунок. По умолчанию оба свойства равны 1. Рекомендуемая область значений обоих свойств от -32,767 до 32,767.
- Свойство **Delay** (задержка) определяет время в миллисекундах, через которое последовательно возникают события Change, если пользователь непрерывно щелкает кнопку прокрутки или левую кнопку мыши, указывающей на полосу прокрутки. По умолчанию устанавливается значение в 50 миллисекунд.
- Свойство **ProportionalThumb** определяет размер бегунка: True — размер бегунка пропорционален размеру области прокрутки (это значение по умолчанию); False — система определяет фиксированный размер бегунка.

SpinButton — счетчик

SpinButton (счетчик, ворот) позволяет пользователю увеличивать и уменьшать числовую характеристику до тех пор, пока он не установит требуемое значение. Один щелчок кнопки прокрутки увеличивает или уменьшает значение свойства Value на величину, заданную свойством **SmallChange**. Как и для ScrollBar, интервал изменения числовой характеристики определяется значениями свойств Min и Max, вертикальная или горизонтальная ориентация счетчика — свойством Orientation, а задержка между повторными событиями Change — свойством Delay.

Чтобы изменения Value были видны пользователю, счетчик надо связать с полем ввода или с меткой в процедуре обработки события Change так же, как для полосы прокрутки.

TextBox — поле ввода (окно редактирования)

Элемент TextBox (поле ввода) — основное средство для ввода пользователем числовых и текстовых данных в систему. Его можно использовать и для вывода информации для пользователя. Поле ввода представляет собой простой одно- или многострочный текстовый редактор, имеющий, кроме указанных выше общих свойств объектов-элементов управления, целый ряд специфических свойств.

- Свойство **Text** содержит текст, находящийся в поле ввода (то же значение имеет и свойство Value).
- Булево свойство **MultiLine** определяет, сколько редактор содержит строк: одну или несколько. False (по умолчанию) — одна строка; True — допускается много строк.
- Переход на новую строку при этом регулируется булевым свойством **WordWrap**. Если оно равно True, текст автоматически переносится на

новую строку при достижении правой границы поля. При значении `False` новая строка в редакторе появляется лишь при обнаружении или при вставке в текст символа перехода на новую строку [клавиши `Ctrl+Enter` или `Shift+Enter` — при вводе и `Chr(13)` или `Chr(10)` — при выводе]. Иначе при достижении правой границы поля текст сдвигается влево, и ввод продолжается в той же строке. Параметр `WordWrap` влияет на работу только многострочного редактора (т. е. при `MultiLine = True`).

- Булево свойство **AutoSize** заставляет поле автоматически менять размер, подстраивая его под текст. Его имеет смысл устанавливать в `True` для полей, используемых для вывода информации (например, для полей с многострочными заголовками или сообщениями).
- Свойство **MaxLength** задает максимальную допустимую длину текста в поле ввода. По умолчанию равно 0, что означает отсутствие ограничений (точнее, ограничения накладываются лишь доступной памятью).
- Свойство **TextLength** возвращает текущий размер текста в поле (для многострочного текста учитываются также символы перевода строки), а **LineCount** — текущее количество строк в тексте.
- Свойство **CurLine** устанавливает или возвращает номер строки редактора, в которой находится курсор. Строки нумеруются с 0. Свойство **CurX** задает текущую горизонтальную позицию курсора, а свойство **CurTargetX** совпадает с `CurX` при работе в текущей строке, но сохраняет старое значение при переходе на новую строку в результате нажатия клавиш-стрелок, пролистывания или при вставке новой строки. Эти свойства позволяют управлять положением курсора при перемещении по строкам редактора. Единицей измерения для обоих свойств служит 0.01 см. Все три свойства работают, только когда поле ввода находится в фокусе.
- Свойство **SelText** возвращает текст, выделенный в поле ввода, свойство **SelLength** возвращает его длину, а **SelStart** определяет начальную позицию выделенного текста в поле, а если такого текста нет, то позицию курсора. Значения двух последних свойств измеряются в символах и могут лежать в интервале от 0 до общего числа символов в поле. Доступ ко всем трем свойствам возможен всегда независимо от того, находится ли поле ввода в фокусе. Изменение значения свойства `SelStart` приводит к снятию выделения с текста (`SelText` становится пустой строкой), установке курсора в новую позицию и установке `SelLength` в 0.
- Поле ввода можно использовать для ввода паролей и других данных, которые не должны появляться на экране, когда пользователь набирает их на клавиатуре. Эта функция поддерживается свойством **PasswordChar**, задающим символ-заместитель, который появляется

Лабораторная работа №1. Пользовательские функции VBA.

1. Запустите электронную таблицу Microsoft Excel.
2. Запустите редактор VBA¹.
Выполните команду Сервис – Макрос – Редактор Visual Basic. В результате вы попадете в интегрируемую среду разработки приложений IDE редактора Visual Basic (рис. 1).

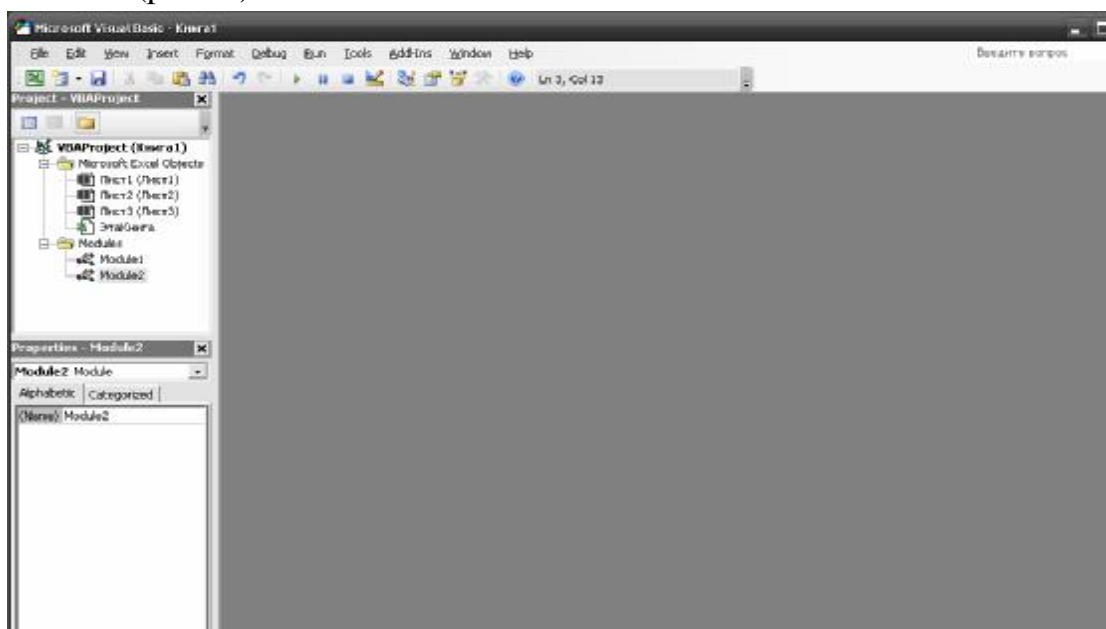


Рис.1. Редактор Visual Basic.

3. Построим функцию, которая возвращает стоимость товара по его стоимости без НДС и значению ставки НДС.

Выполните команду Insert – Module, и наберите код:

```
Function Стоимость (СтоимостьБезНДС, НДС)  
Стоимость = СтоимостьБезНДС * (1 + НДС / 100)  
End Function
```

4. С панели задач переключитесь на Microsoft Excel. Создайте следующую таблицу:

¹ VBA (Visual Basic for Applications) – единый для всех приложений Microsoft Office язык программирования, построенный на основе Visual Basic.

Стоимость без НДС	НДС	Стоимость
1000	25	

5. Выделите ячейку C2, в которой найдем значение функции.
6. Выполним команду Вставка – Функция.
7. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Стоимость и нажмите кнопку ОК.
8. В поле СтоимостьБезНДС введите ссылку на ячейку A2, а в поле НДС – ссылку на ячейку B2 и нажмите кнопку ОК В результате в ячейке C2 появиться значение 1250.
9. Сохраните созданную работу в рабочей папке.

Лабораторная работа №2.

Разработка функций для реализации линейных и разветвляющихся алгоритмов.

1. Запустите электронную таблицу Microsoft Excel.
2. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор Visual Basic. В результате вы попадете в интегрируемую среду разработки приложений IDE редактора Visual Basic.

3. Построим функцию вычисления выражения $y = \frac{x^2 - 5\sqrt{2}}{2x^3 + 1}$

Выполните команду Insert – Module, и наберите код:

```
Public Function fun1(x)
fun1=(x*x-5*2^0.5)/(2*x^3+1)
End Function
```

С панели задач переключитесь на Microsoft Excel. Выполним команду Вставка – Функция. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Fun1 и нажмите кнопку ОК. В появившемся окне в поле X введите, например 6, Y=0,06681.

4. Построим функцию вычисления полупериметра треугольника по трем сторонам a,b,c. В редакторе VBA выполните команду Insert – Module, и наберите код:

```
Public Function Полупериметр(a, b, c)
Полупериметр=(a+b+c)/2
End Function
```

С панели задач переключитесь на Microsoft Excel. Установите курсор в ячейку A2. Выполним команду Вставка – Функция. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Полупериметр и нажмите кнопку ОК. В появившемся окне в поле А, В, С введите, например 2,3,4, Результат – 4,5.

5. Построим функцию вычисления длины окружности и площади круга заданного радиуса R. В редакторе VBA выполните команду Insert – Module, и наберите код:

```
Public Function Окружность(R)
    Pi=3.14
    a=2*Pi*R
    b=Pi*R^2
    Окружность="C="+str(a)+" S="+str(b)
End Function
```

С панели задач переключитесь на Microsoft Excel. Установите курсор в ячейку A3. Выполним команду Вставка – Функция. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Окружность и нажмите кнопку ОК. В появившемся окне в поле R введите любое значение.

6. Построим функцию нахождения максимального элемента из трех чисел a, b, c. В редакторе VBA выполните команду Insert – Module, и наберите код:

```
Public Function Max(a, b, c)
    If a > b Then
        m = a
    Else
        m = b
    End If
    If c > m Then
        Max = c
    Else
        Max = m
    End If
End Function
```

С панели задач переключитесь на Microsoft Excel. Установите курсор в ячейку A4. Выполним команду Вставка – Функция. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Max и нажмите кнопку ОК. В появившемся окне введите любые значения.

7. Построим функцию нахождения корней квадратного уравнения. В редакторе VBA выполните команду Insert – Module, и наберите код:

```
Public Function Корни(a, b, c)
    d = b ^ 2 - 4 * a * c
```

```

If d >= 0 Then
    x1 = (-b + d ^ (1 / 2)) / (2 * a)
    x2 = (-b + d ^ (1 / 2)) / (2 * a)
    Корни = "x1=" + str(x1) + "; x2=" + str(x2)

```

```
Else
```

```
    Корни = "корней нет"
```

```
End If
```

```
End Function
```

С панели задач переключитесь на Microsoft Excel. Установите курсор в ячейку A5. Выполним команду Вставка – Функция. В окне Мастера функций в списке Категория найдем значение Определенные пользователем, в списке Функция выберите функцию Корни и нажмите кнопку ОК. В появившемся окне введите любые значения.

8. Самостоятельно создайте пользовательскую функцию вычисления выражения $y = \frac{2x+a}{b-1} + \sqrt[3]{|x+a|}$

9. Самостоятельно создайте пользовательскую функцию для нахождения минимального числа из трех заданных чисел А, В, С.

10. Сохраните созданную работу в рабочей папке.

Лабораторная работа №3. Расчет комиссионных.

1. Запустите электронную таблицу Microsoft Excel.
2. Запустите редактор VBA.

Выполните команду Сервис – Макрос – Редактор Visual Basic.

3. Нам необходимо разработать функцию, позволяющую рассчитывать комиссионные. Процент комиссионных зависит от объема проданного товара и начисляется по следующему правилу, представленному в табл. 1.

Таблица 1. Правила расчета комиссионных

Объем продаж за неделю, руб.	Комиссионные, %
От 0 до 9999	8
От 10000 до 19999	10
От 20000 до 39999	12
Более 40000	14

4. Построим для расчета в стандартном модуле пользовательскую функцию.

Выполним команду Insert – Module, и наберите код:

```

Function Комиссионные(Продажи)
If Продажи <= 9999 Then

```



```

Комиссионные = Продажи * 0.08
ElseIf Продажи <= 19999 Then
Комиссионные = Продажи * 0.1
ElseIf Продажи <= 39999 Then
Комиссионные = Продажи * 0.12
Else
Комиссионные = Продажи * 0.14
End If
End Function

```

5. С панели задач переключитесь на Microsoft Excel. Создайте следующую таблицу и рассчитайте комиссионные:

Сотрудник	Объем продаж за неделю, руб.	Комиссионные, руб.
Иванов	20000	
Петров	1000	
Сидоров	30000	
Смирнов	40000	
Пенкина	10000	
Федорова	32500	
Федорчук	41000	
Якин	32000	
Яшин	35000	

6. Для расчета Комиссионных выполните команду Вставка – Функция, Категория -

Определенные пользователем, в списке Функция выберите функцию Комиссионные и нажмите кнопку ОК. В появившемся окне в поле Продажи вводите соответствующее число.

7. Усложним задачу. Будем считать, что комиссионные зависят от ставки, занимаемой менеджером. Если он принят в постоянный штат фирмы, то комиссионные начисляются по описанному выше закону (таб. 1). Если же он находится на испытательном сроке, то его комиссионные составляют 75% от номинала.

8. Вставьте столбец Ставка между Объем продаж за неделю и Комиссионные. Заполните столбец: если испытательный срок Ставка – 0, если штатный – 1.

Сотрудник	Объем продаж за неделю, руб.	Ставка	Комиссионные, руб.
Иванов	20000	0	
Петров	1000	1	
Сидоров	30000	1	
Смирнов	40000	1	
Пенкина	10000	0	
Федорова	32500	1	

Федорчук	41000	1	
Якин	32000	0	
Яшин	35000	0	

9. Внесите изменения в функцию:

```
Function Комиссионные (Продажи, Ставка)
If Продажи <= 9999 Then
Оплата = Продажи * 0.08
ElseIf Продажи <= 19999 Then
Оплата = Продажи * 0.1
ElseIf Продажи <= 39999 Then
Оплата = Продажи * 0.12
Else
Оплата = Продажи * 0.14
End If
If Ставка = 0 Then
Комиссионные = 0.75 * Оплата
Else
Комиссионные = Оплата
End If
End Function
```

10. Рассчитайте новые значения столбца Комиссионные, используя Мастер функций.

(Например, для Иванова Комиссионные = 1800).

11. Сохраните созданную работу в рабочей папке.

Лабораторная работа №4. Построение диаграммы для функции.

1. Запустите электронную таблицу Microsoft Excel.
2. Построим диаграмму для функции $y = x^2 - 3x$.
3. В ячейку A1 введём x , в B1 – y
4. С помощью арифметической прогрессии в столбец A вводим значения x , для этого заполним ячейку A2 (к примеру, $A2 = -1$), затем выделим область A2-A10 и нажмём Правка – Заполнить – Прогрессия. В появившемся окне поставим: флажок расположение – по столбцам, тип – арифметическая, шаг – 0,5.
5. В ячейку B2 вводим формулу $x^2 - 3x$ ($= A2^2 - 3 * A2$).
6. Заполним ячейки от B2 до B10, перетаскивая зажатый курсор мыши
7. Построим диаграмму для заданной функции, для этого выполним команду Сервис – Макрос – Редактор Visual Basic
8. В появившемся окне нажмём View – Code и введём код

Sub CreateChart()

' создаем пустой лист диаграммы'с именем диаграмма 1, расположенный последним

ActiveWorkbook.Charts.Add Worksheets(Worksheets.Count)

' задаем параметры диаграммы с помощью метода ChartWizard

Charts(Charts.Count).ChartWizard Source:=Worksheets(1).Range("a2:b8"),

gallery:=xl3DColumn, PlotBy:=xlColumns, CategoryLabels:=1, SeriesLabels:=1,

Title:="Диаграмма функции", HasLegend:=True

End Sub

7. Проект готов. Для проверки нажмите F5, будет построена диаграмма.(см. рис.1)

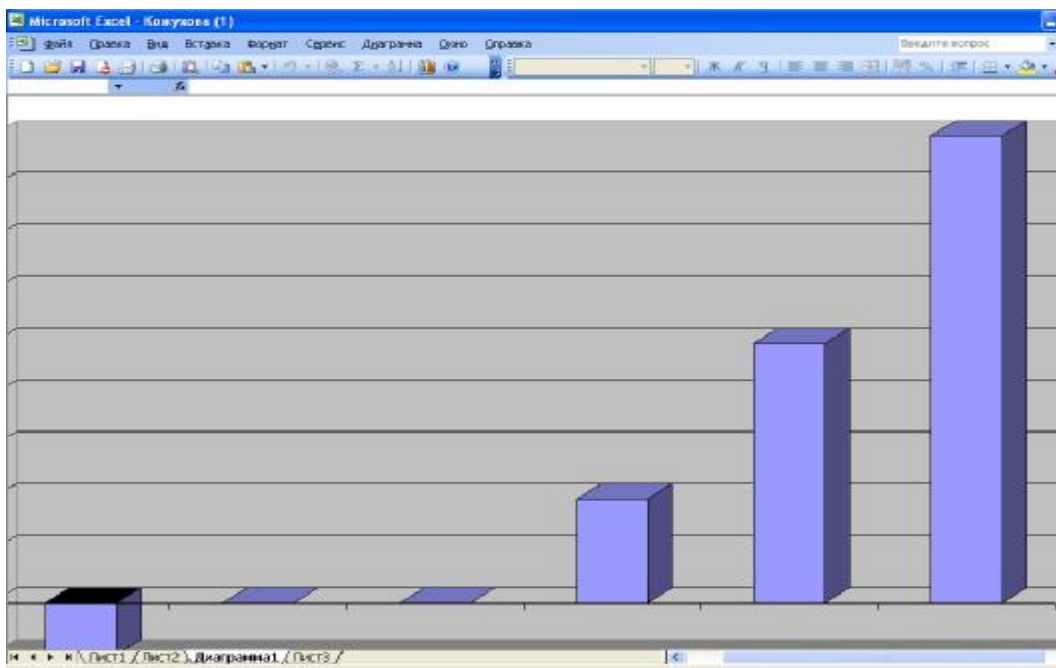


Рис.1. Построенная диаграмма.

9. Сохраните работу в своей рабочей папке.

10. Самостоятельно постройте диаграмму для функции $y = x^3 - 2x^2 + 4$.

Лабораторная работа №5.

Элементы управления.

1. Запустите электронную таблицу Microsoft Excel.
2. Выполните команду Вид – Панели инструментов – Элементы управления.
3. Создадим кнопку.

Нажмите кнопку Кнопка на панели инструментов Элементы управления.
(см. рис. 1)

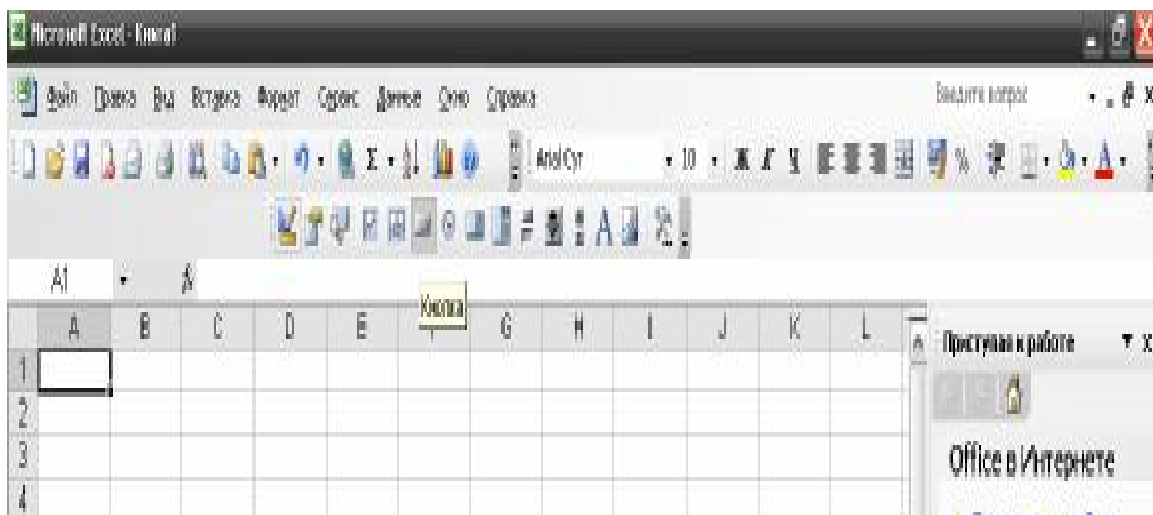


Рис.1. Панель элементов управления.

4. В любом месте рабочего листа прорисуйте кнопку произвольного размера точно так же, как и при помощи любого графического редактора.
5. На панели Элементы управления нажмите кнопку Свойства. На экране отобразится окно Properties. Установите в окне свойства Name вместо CommandButton1 – cmdПривет. Измените значение свойства Caption – Привет.
6. Нажмите кнопку Исходный текст на панели инструментов Элементы управления. В результате откроется редактор Visual Basic, причем в окне кода автоматически будет создана первая и последняя инструкция обработки события кнопки.
7. Добавьте следующую инструкцию:
MsgBox "Привет, всем"
8. Проект готов. Нажмите на кнопку, и если на экране отобразится диалоговое окно с приветствием, то вы сделали все правильно.
9. Создайте самостоятельно аналогичным способом Кнопку2 - Приветствие, при нажатии на которую появлялось сообщение: Привет от 511 группы.
10. Сохраните созданную работу в рабочей папке.

Лабораторная работа №6. Создание кнопочного сценария расчета.

1. Запустите электронную таблицу Microsoft Excel.
2. Мы попытаемся предсказать суммарную прибыль, однако у нас нет точных сведений о том, какова будет почасовая оплата труда, стоимость единицы материала, количество используемого материала. Эти данные определяются двумя возможными вариантами.(см. таб. 1)

	Первый вариант	Второй вариант
Почасовая оплата, руб.	23	29
Стоимость материала, руб.	62	55
Количество материала, шт.	5	3

3. Для создания данного проекта в ячейки рабочего листа надо ввести постоянные значения и расчетные формулы.

Ячейка	Значение или формула
B3	12
B5	=B4*B2+B1*B3
B6	5%
B7	=B5*(1+B6)
B8	=B7-B5
B9	36
B10	=B8*B9

8. Получится таблица следующего содержания:

Почасовая оплата, руб.	29
Стоимость материала, руб.	55
Норма времени, ч.	12
Количество материала, шт.	3
Себестоимость, руб.	513
Товарная наценка	0,05
Отпускная цена, руб.	538,65
Прибыль на одно изделие, р.	25,65
Количество изделий, шт.	36
Суммарная прибыль	923,4

9. На рабочем листе создадим две кнопки.

Нажмите кнопку Кнопка на панели инструментов Элементы управления. Растяните примерно с D2:G3. На панели Элементы управления нажмите кнопку Свойства. На экране отобразится окно Properties. Установите в окне свойства Name вместо CommandButton1 – cmdFirst. Измените значение свойства Caption – Первый.

Аналогично создайте вторую кнопку. Место положение D5:G6. Свойства Name – cmdSecond, Caption – Второй.

6. Нажмите кнопку Исходный текст на панели инструментов Элементы управления.

В результате откроется редактор Visual Basic.

7. В модуле рабочего листа наберите следующий код.

```
Private Sub cmdFirst_Click()  
Costs 23, 62, 5
```

```

End Sub
Private Sub cmdSecond_Click()
Costs 29, 55, 3
End Sub
Private Sub Costs(job, thing, item)
Range("B1").Value = job
Range("B2").Value = thing
Range("B4").Value = item
End Sub

```

10. Проект готов. Проверьте работу наших кнопок. Обратите внимание, что при нажатии на кнопки меняются числа и значения суммарной прибыли (см. рис.1).

	A	B	C	D	E	F	G	H	I	J
1	Почасовая оплата, руб.	29								
2	Стоимость материала, руб.	55	Первый							
3	Норма времени, ч.	12								
4	Количество материала, шт.	3								
5	Себестоимость, руб.	513	Второй							
6	Товарная наценка	0,05								
7	Отпускная цена, руб.	538,65								
8	Прибыль на одно изделие, р.	25,65								
9	Количество изделий, шт.	36								
10	Суммарная прибыль	923,4								
11										
12										
13										
14										

Рис.1. Расчет суммарной прибыли – второй вариант.

11. Сохраните созданную работу в рабочей папке.

Лабораторная работа №7.

Автоматизация расчета эффективности капиталовложений с помощью счетчика.

1. Запустите электронную таблицу Microsoft Excel.
2. Создайте таблицу следующего содержания:

Размер ссуды	10 000р.
Срок	6 лет
Ежегодно возвращаемые деньги	2 000р.
Годовая процентная ставка	6%
Чистый текущий бъем вклада	
Вывод	

3. Ячейку B5 рассчитать по формуле ПС. (Вставка – Функция, категория Финансовые) Заполните окно как на рис. 1.

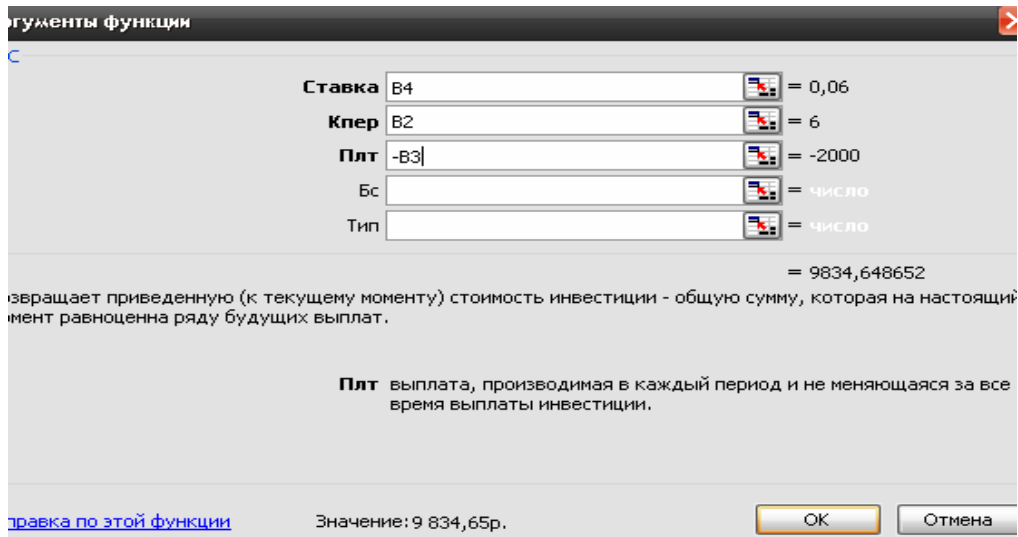


Рис1. Форма для заполнения.

4. В ячейку B6 введите формулу: =ЕСЛИ(B1<B5;"Выгодно дать деньги в долг"; ЕСЛИ(B5=B1;"Варианты равносильны";"Выгоднее деньги положить под проценты"))
5. Создайте на рабочем листе в ячейке C4 счетчик (воспользуйтесь кнопкой Счетчик



на панели инструментов Элементы управления).

6. На панели Элементы управления нажмите кнопку Свойства. На экране отобразится окно Properties. Установите в окне свойства Name вместо SpinButton1 – spnRate.

7. Нажмите кнопку Исходный текст на панели инструментов Элементы управления.

В результате откроется редактор Visual Basic. В модуле рабочего листа наберите код:

```
Private Sub spnRate_Change()
Range("B4").Value = 0.01 * spnRate.Value
End Sub
Private Sub Worksheet_Change(ByVal Target As Range)
If Target.Address = "$B$4" Then
spnRate.Value = CInt(100 * Range("B4").Value)
End If
End Sub
```

8. В результате получится следующее:

Размер ссуды	10 000р.
Срок	6 лет
Ежегодно возвращаемые деньги	2 000р.

Годовая процентная ставка 7%

Чистый текущий объем вклада 9 533,08р.

Вывод Выгоднее деньги положить под проценты

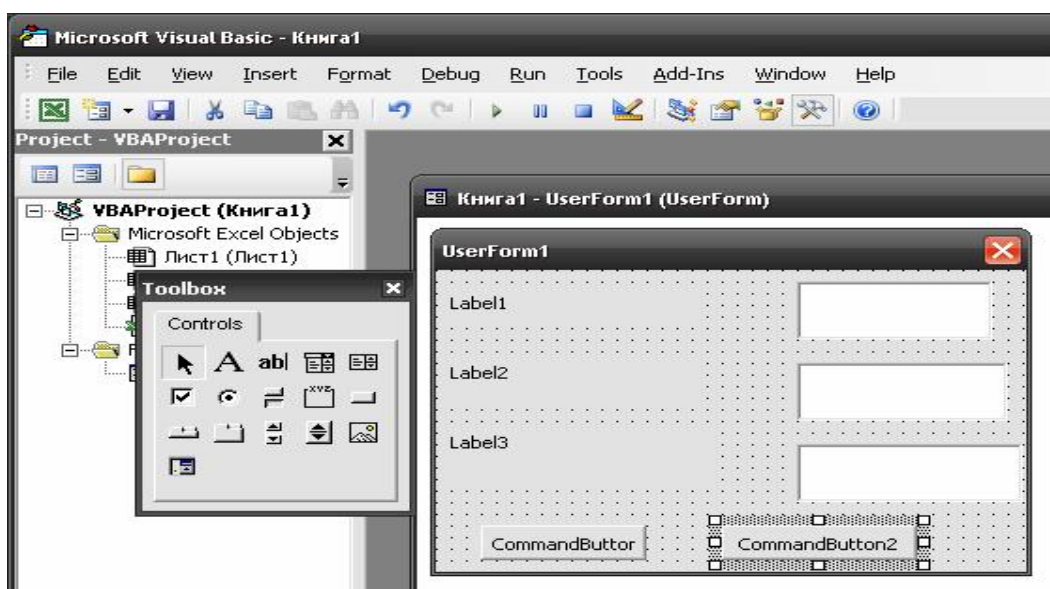
9. Проверьте работоспособность счетчика. При нажатии на счетчик должна меняться

Годовая процентная ставка и соответственно Чистый текущий объем вклада. Например при ставке 13% - объем вклада = 7995,10 р.

10. Сохраните созданную работу в рабочей папке.

Лабораторная работа №8. Создание пользовательской формы.

1. Создадим диалоговое окно, в котором производится расчет стоимости товара с учетом НДС по его стоимости без НДС и значению ставки НДС.
2. Запустите электронную таблицу Microsoft Excel.
3. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор Visual Basic.
4. Добавьте форму. Выберите команду Insert – UserForm.
5. На ней расположите 3 надписи, 3 поля ввода данных, 2 кнопки. Выберите на Панели элементов кнопку Надпись (Label 1), Поле ввода (TextBox1), Кнопка (CommandButton1). Должно получиться следующее (рис.6):



- Рис.6. Окно с расположенными на нем элементами управления
6. Теперь при помощи окна Свойств Properties задайте элементам управления

следующие значения таб. 6.(для вызова окна нажмите клавишу F4, для изменения свойства надписей, полей ввода и кнопок надо их выделить и нажать правую кнопку мыши выбрать меню Properties).

таб. 6. Значение свойств

Элемент управления	Свойства	Значение
Форма	Caption	Расчет стоимости
Надпись (Label1)	Caption	Стоимость без учета НДС
Надпись (Label2)	Caption	НДС
Надпись (Label3)	Caption	Стоимость с учетом НДС
Кнопка(CommandButton1)	Name Caption	cmdOK OK
Кнопка(CommandButton2)	Name Caption	cmdCancel Cancel
Поле ввода(TextBox1)	Name	txtCost
Поле ввода(TextBox2)	Name	txtTax
Поле ввода(TextBox3)	Name	txtResult

7. В модуле формы наберите следующий код (в поле формы нажмите правую кнопку мыши и выберите меню View Code)

```
Private Sub cmdOK_Click()
Dim dCost As Double
Dim iTax As Long
Dim dResult As Double
iCost = CDbl(txtCost.Text)
iTax = CLng(txtTax.Text)
dResult = iCost * (1 + iTax / 100)
dResult = Format(dResult, "Fixed")
txtResult.Text = CStr(dResult)
End Sub
Private Sub cmdCancel_Click()
Unload Me
End Sub
```

8. Проект готов. Нажмите клавишу F5, на экране отобразится диалоговое окно Расчет стоимости (см рис.6.1.). В поле Стоимость без учета НДС введите 1000, в поле НДС – 25, нажмите ОК. В поле Стоимость с учетом НДС будет выведена искомая стоимость. Для закрытия окна нажмите кнопку Cancel.

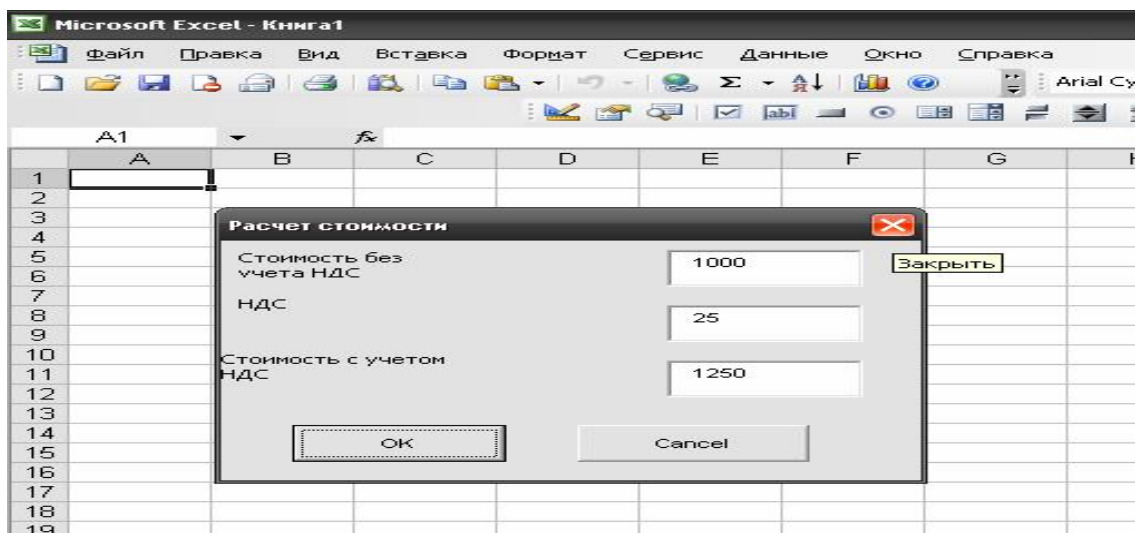


Рис. 6.1. Диалоговое окно Расчет стоимости
12. Сохраните созданную работу в рабочей папке.

Лабораторная работа №9.

Создание пользовательской формы. Пример использования рамки и переключателя.

1. Создадим диалоговое окно, в котором находится сумма либо произведение двух чисел. Исходные данные вводятся в два поля. При выборе первого переключателя после нажатия кнопки ОК будут складываться данные из полей ввода, а при выборе второго – перемножаться, а результат выводится в третье поле.
2. Запустите электронную таблицу Microsoft Excel.
3. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор Visual Basic.
4. Добавьте форму. Выберите команду Insert – UserForm.
5. На ней расположите 3 поля ввода данных, кнопку и рамку, внутри которой расположатся два переключателя.
6. Выберите на Панели элементов кнопку: Поле ввода (TextBox1), Кнопка (CommandButton1), Рамку (Frame) и Переключатель (OptionButton). Должно получиться следующее (рис.7):

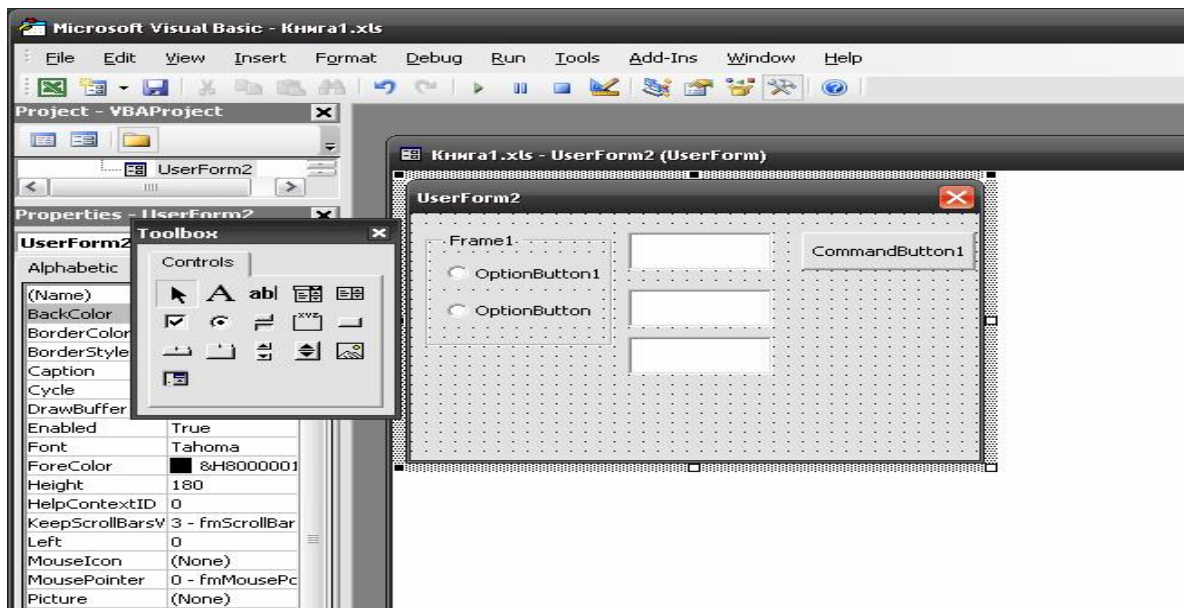


Рис. 7. Окно с расположенными на нем элементами управления.

7. При помощи окна Properties установите значения свойств Name и Caption полей элементов управления из таб.7. Для этого выделите элемент и нажмите правую кнопку мыши, из меню выберите Properties.

Таб.7. Значения свойств

Поле ввода(TextBox1)	Name	txtItem1
Поле ввода(TextBox2)	Name	txtItem2
Поле ввода(TextBox3)	Name	txtResult
Кнопка(CommandButton1)	Name Caption	cmdOK OK
Рамка(Frame1)	Caption	Операция
Переключатель	Name Caption	optAdd Сложение
Переключатель	Name Caption	optMult Произведение

8. В модуле формы наберите следующий код (для этого нажмите правую кнопку мыши в пределах формы и выберите меню View Code)

```
Private Sub UserForm_Initialize()
    optAdd.Value = True
    txtResult.Enabled = False
End Sub
Private Sub cmdOK_Click()
    Dim a As Double, b As Double
    a = Cdbl(txtItem1.Text)
    b = Cdbl(txtItem2.Text)
    If optAdd.Value Then txtResult.Text = a + b
    If optMult.Value Then txtResult.Text = a * b
End Sub
Private Sub optAdd_Click()
```

```

Caption = "Сложение"
End Sub
Private Sub optMult_Click()
Caption = "Произведение"
End Sub

```

9. Проект готов. Нажмите клавишу F5, на экране отобразится диалоговое окно Расчет стоимости (см рис.7.1.)

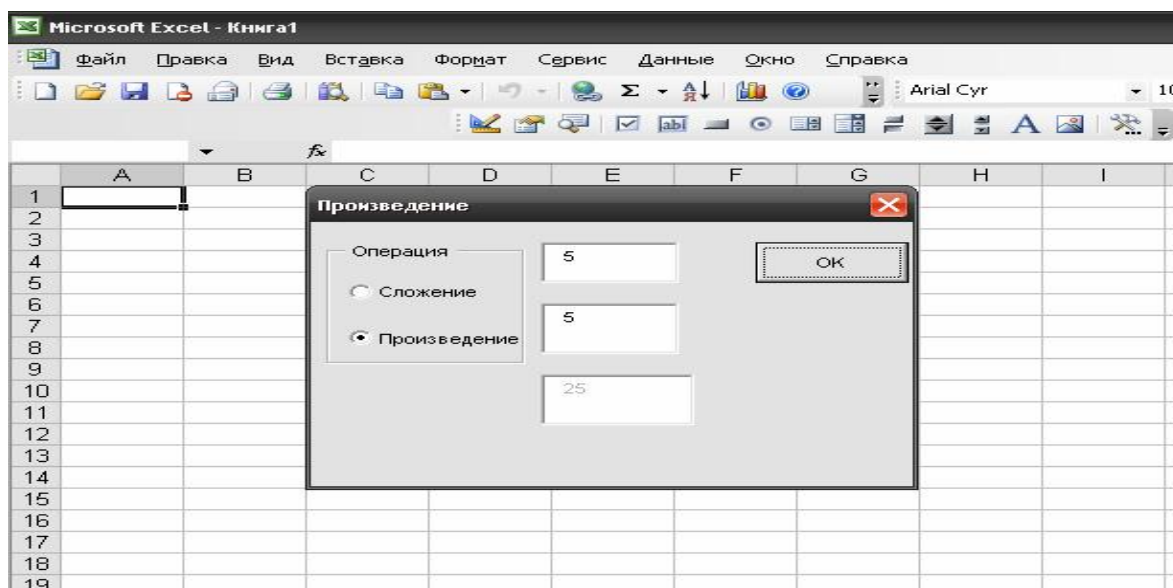


Рис. 7.1. Сумма или произведение двух чисел.

10. Сохраните созданную работу в рабочей папке.

Лабораторная работа №10.

Создание пользовательской формы. Пример использования флажков.

1. Создадим проект для решения следующего типа задач по расчету ренты.

Предположим, что выкупается страховка, по которой выплачивается по 500р. В конце каждого месяца в течение 20 последующих лет. Стоимость ренты составляет 60000р. И выплачиваемые деньги принесут 8% годовых.

2. Запустите электронную таблицу Microsoft Excel.

3. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор VB.

4. Добавьте форму. Выберите команду Insert – UserForm.

На ней расположите (рис. 8.) 5 полей ввода данных, 5 полей надписей, кнопку и флажок.

5. Выберите на Панели элементов кнопку: Поле ввода (TextBox1), Надпись (Label 1), Кнопка (CommandButton1), Флажок (CheckBox). Должно получиться следующее:

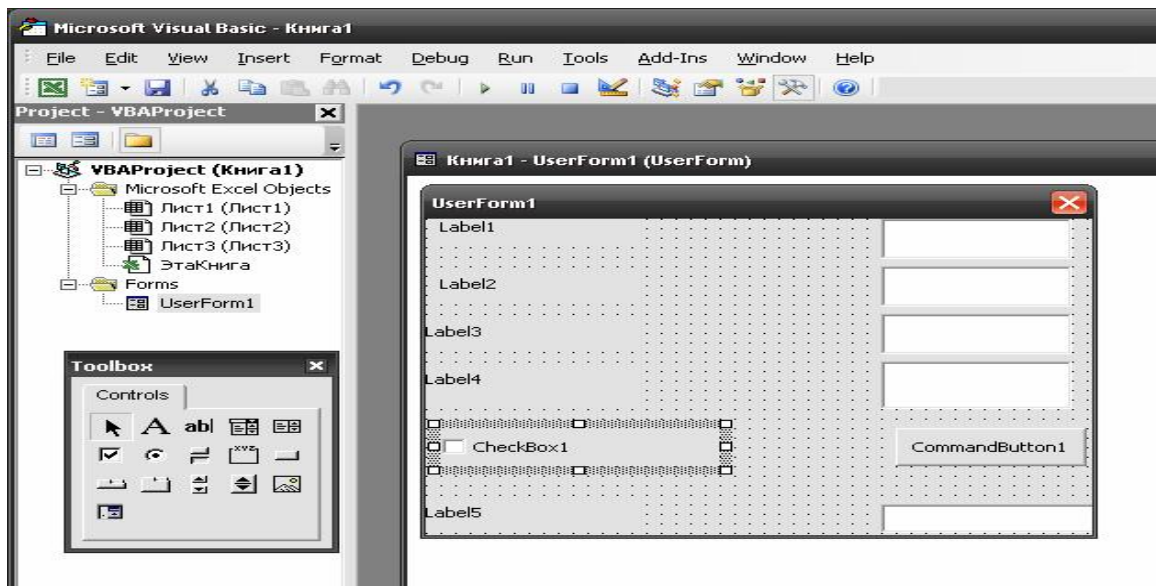


Рис. 8. Окно с расположенными на нем элементами

6. При помощи окна Properties установите значения свойств Name и Caption полей элементов управления из таб.8. Для этого выделите элемент и нажмите правую кнопку мыши, из меню выберите Properties.

Таб. 8. Значение свойств

Форма (для вызова окна свойства Properties нажмите F4)	Caption	Рента
Надпись	Caption	Стоимость ренты
Поле ввода	Name	txtPV
Надпись	Caption	Процентная ставка, годовых
Поле ввода	Name	txtRate
Надпись	Caption	Разовая выплата
Поле ввода	Name	txtPmt
Надпись	Caption	Срок, лет
Поле ввода	Name	txtNPer
Надпись	Caption	Настоящий объем вклада
Поле ввода	Name	txtResult
Кнопка	Name	cmdOK
	Caption	OK
Флажок	Name	chkType
	Caption	Выплата в конце каждого месяца

7. В модуле формы наберите следующий код (для этого нажмите правую кнопку мыши в пределах формы и выберите меню View Code)

```
Private Sub UserForm_Initialize()
    ChkType.Value = True
End Sub
Private Sub cmdOK_Click()
    Dim dRate, dNper, dPmt, bType, dResult
```

```

dRate = CDBl(txtRate.Text)
dNper = CDBl(txtNPer.Text)
dPmt = CDBl(txtPmt.Text)
If ChkType.Value Then
bType = 0
Else
bType = 1
End If
dResult = -PV(dRate / (12 * 100), dNper * 12, dPmt, , bType)
txtResult.Text = CStr(Format(dResult, "Fixed"))
End Sub
Private Sub chkType_Change()
With ChkType
If .Value Then
Caption = "Рента: выбрана выплата в конце каждого месяца"
Else
Caption = "Рента: выбрана выплата в начале каждого месяца"
End If
End With
End Sub

```

8. Проект готов. Нажмите клавишу F5, на экране отобразится диалоговое окно Рента (см. рис.8.1.). Введите заданные значения. Проверьте работоспособность формы, меняя значения ставки или срока. Обратите внимание, что если флажок установлен, то в заголовке окна появился текст "Рента: выбрана выплата в конце каждого месяца", а если сброшен, то "Рента: выбрана выплата в начале каждого месяца". Таким образом, флажок в данном проекте не только фиксирует тот или иной выбор, но и управляет заголовком окна.

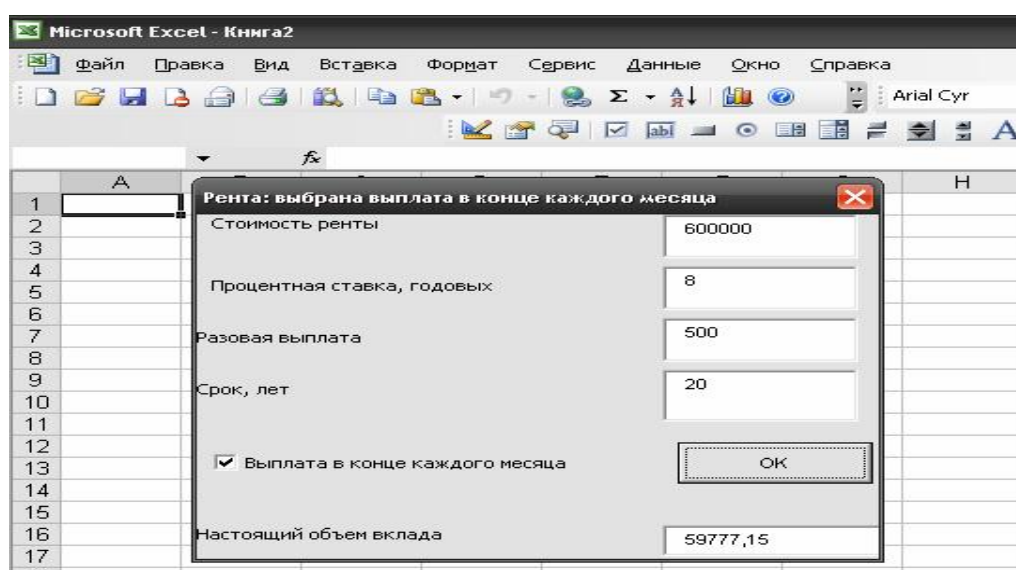


Рис.8 Окно Рента.

Лабораторная работа №11.

Пример создания диалогового окна

1. Давайте создадим диалоговое окно «Цветная форма», в котором будут расположены изображение (Image), поле ввода (TextBox) и командная кнопка (CommandButton).

2. Запустите электронную таблицу Microsoft Excel.

3. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор VB

4. В меню *Вставка* выберите команду UserForm для создания нового диалогового окна.

5. Щелкнув правой кнопкой появившееся окно UserForm1, выберите в контекстном меню команду *Свойства* (Properties). Введите новое имя ColorForm1 в правом столбце строки со свойством (Name), в качестве значения свойства Caption введите заголовок окна: «Цветная форма». Щелкните правый столбец в строке со свойством BackColor (Цвет фона), затем — появившийся справа значок списка. В таблице появившегося окна щелчком выберите понравившийся цвет. Окно «перекрасится». Закройте окно свойств.

6. Перетащите с панели инструментов на окно элемент Image (изображение).

Щелкните его правой кнопкой и снова выберите команду *Свойства*. В появившемся окне со списком свойств отыщите строку со свойством Picture (Рисунок). Чтобы перейти к поиску нужного файла с картинкой, щелкните кнопку с многоточием (...) справа в этой строке. В диалоговом окне *Загрузка рисунка* (Load Picture) выберите файл с симпатичной картинкой и щелкните кнопку ОК.

7. Перетащите на диалоговое окно элемент управления поле ввода (TextBox). Как и в п. 5, щелкнув его правой кнопкой, вызовите окно *Свойства*. Введите в качестве значения свойства Value название выбранной вами картинки — оно появится в окне редактора. Затем выберите свойство Font Справа появится кнопка (...). Щелкните ее и в появившемся списке шрифтов выберите шрифт для редактора.

8. Перетащите на диалоговое окно командную кнопку (CommandButton). Вызовите окно *Свойства* для этой кнопки, и измените в нем значение свойства Caption (Заголовок) на «Нажми меня» — текст станет именем кнопки. Измените имя кнопки (Name) на cmdClickMe. По этому имени к кнопке будут обращаться процедуры и методы в программе. Введите справку «Командная кнопка» как значение свойства ControlTipText — надпись будет появляться на экране под кнопкой всякий раз, когда на ней окажется указатель. Установите для кнопки ключ быстрого выбора: найдите в списке свойств строку со свойством Accelerator (Ускоритель) и введите в поле справа букву «Н» — в имени кнопки первый символ «Н» будет подчеркнут и нажатие клавиш Alt+N будет эквивалентно выбору кнопки. Установите

также подходящий шрифт с помощью свойства Font, как и для поля ввода.

9. В меню Run (Запуск) выберите команду «Запуск подпрограммы Run Sub/UserForm» или нажмите F5. На экране приложения появится спроектированное диалоговое окно.

10. Щелкните кнопку Close на заголовке диалогового окна, чтобы выйти из него.

11. Сохраните созданную работу в рабочей папке.

Лабораторная работа №12

Табличная база данных туристической фирмы.

1. Создадим проект с табличной базой данных туристической фирмы.
2. Запустите электронную таблицу Microsoft Excel, состоящую из двух рабочих листов.
3. Переименуйте листы в Туры и База данных.
4. В ячейку диапазона A1:A9 листа Туры введите список предлагаемых фирмой туров.

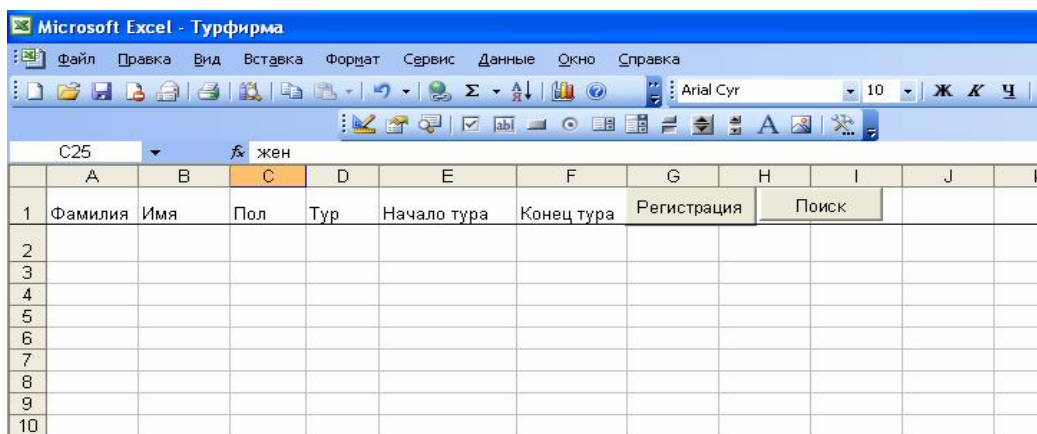
Лондон
Париж
Нью-Йорк
Прага
Осло
Хельсинки
Копенгаген
Берлин
Цюрих

5. В ячейки диапазона A1:F1 листа База данных введите название полей табличной базы данных.

				Начало	
Фамилия	Имя	Пол	Тур	тура	Конец тура

6. Для того чтобы название полей всегда отображалось на экране при вертикальной прокрутке листа, выберите вторую строку листа База данных, а затем укажите команду Окно – Закрепить область.

7. Создайте на рабочем листе База данных две кнопки.



8. При помощи окна свойств Properties (нажмите правую кнопку мыши) установите им следующие значения свойств:

Кнопка	Name Caption	cmdCheckIn Регистрация
Кнопка	Name Caption	cmdSearch Поиск

9. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор VB.

10. Создайте модуль. Выполните команду Insert – Module. Введите программу

Option Explicit

Public bWin As Boolean

'Если bWin равно True, то окно Регистрация

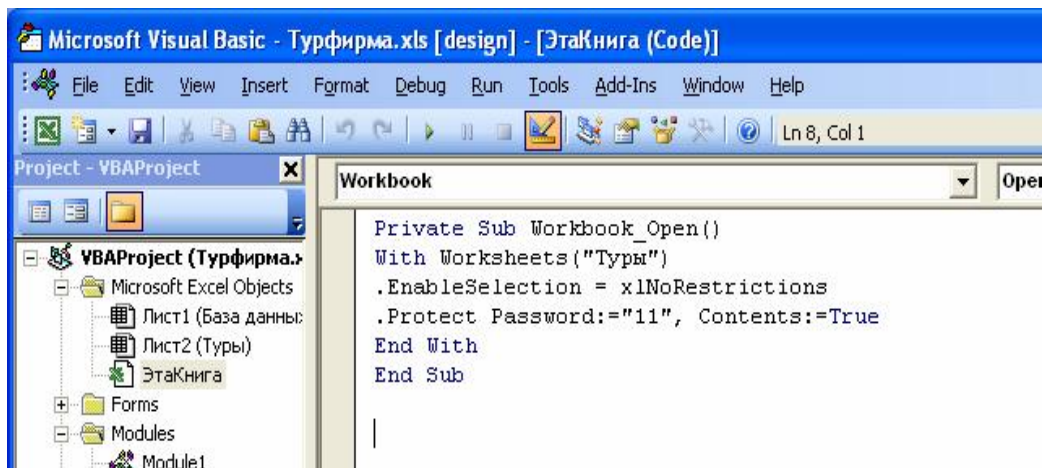
'Если bWin равно False, то окно Редактирование

Public iFound As Integer 'Номер строки с выбранной записью

11. Установим защиту на рабочий лист Туры, но так чтобы в нем возможно было

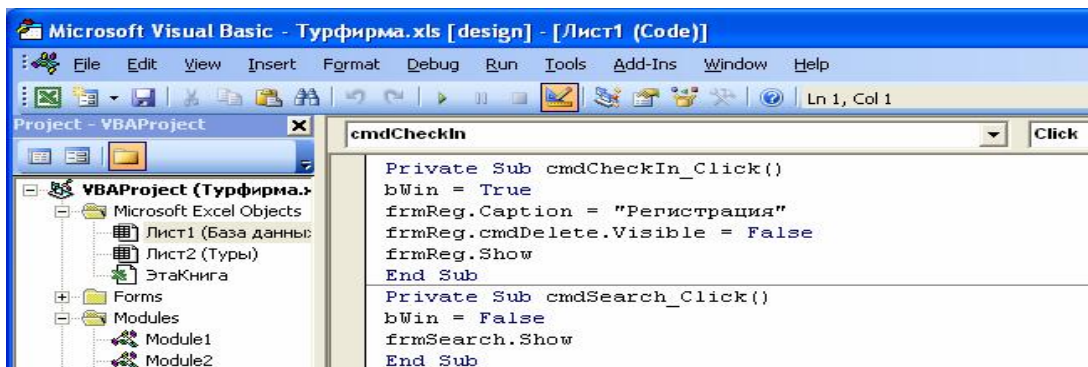
производить выбор диапазонов ячеек. Это нам потребуется, чтобы можно было заполнить список данными, содержащимися на этом рабочем листе.

Для этого, находясь в редакторе VBA, сделайте двойной щелчок на значок ЭтаКнига в левой части экрана. В появившемся модуле введите программу:



```
Private Sub Workbook_Open()
With Worksheets("Турь")
.EnableSelection = xlNoRestrictions
.Protect Password:"11", Contents:=True
End With
End Sub
```

12. В модуле рабочего листа База данных наберите код двух процедур, обрабатывающий событие – нажатие соответствующей кнопки. При нажатии кнопки Регистрация появляется окно Регистрация, а кнопки Поиск – окно Поиск. Для этого, находясь в редакторе VBA, сделайте двойной щелчок на значок Лист1(База данных) в левой части экрана. В появившемся модуле введите программу:



```
Private Sub cmdCheckIn_Click()
bWin = True
frmReg.Caption = "Регистрация"
frmReg.cmdDelete.Visible = False
frmReg.Show
End Sub
Private Sub cmdSearch_Click()
bWin = False
frmSearch.Show
End Sub
```

13. Перейдем к конструированию формы для окон Регистрация и Редактирование. В проекте добавьте форму (Insert - UserForm), на которой расположите пять надписей (Label), четыре поля ввода (TextBox), две рамки (Frame), список (ListBox), два переключателя (OptionButton), три кнопки (CommandButton) и календарь (Calendar) (см. рис 1). Для добавления дополнительных элементов управления на панель элементов необходимо выбрать команду Tools – Additional Controls. В появившемся окне Дополнительные элементы необходимо установить флажок напротив добавляемого элемента. В нашем случае напротив Элемент управления Календарь 11.0. Нажмите ОК. В результате значок выбранного элемента управления появится в панели элементов.

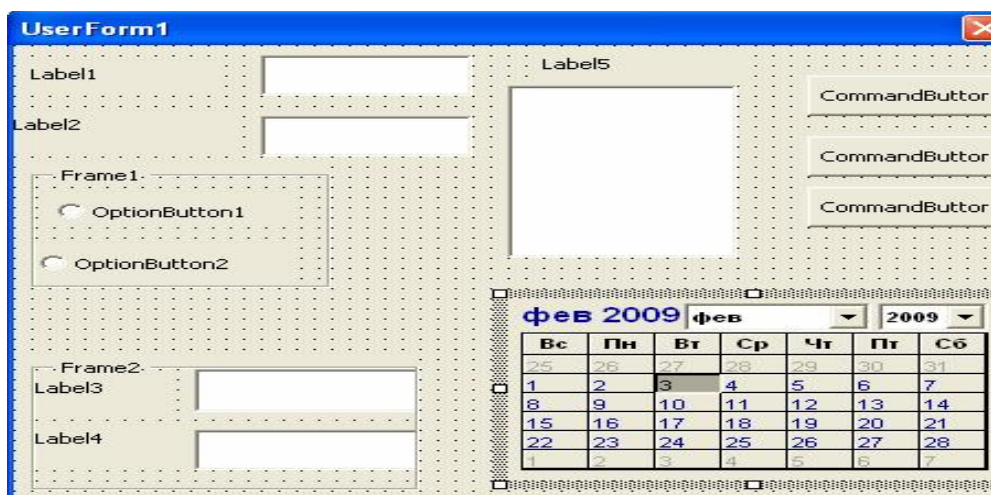


Рис.1. Форма окна Регистрации.

14. При помощи окна Properties установите для них следующие значения свойств:

Форма	Name	frmReg
Кнопка (CommandButton)	Name	cmdSave
	Caption	Сохранить
Кнопка (CommandButton)	Name	cmdCancel
	Caption	Отмена
Кнопка (CommandButton)	Name	cmdDelete
	Caption	Удалить
Надпись (Label1)	Caption	Фамилия
Поле ввода (TextBox)	Name	txtLastName
Надпись (Label2)	Caption	Имя
Поле ввода (TextBox)	Name	txtFirstName
Рамка (Frame)	Caption	Пол
Переключатель (OptionButton)	Name	optMale
	Caption	муж
Переключатель (OptionButton)	Name	optFemale
	Caption	жен
Рамка (Frame)	Caption	Даты тура
Надпись (Label3)	Caption	Начало

Поле ввода (TextBox)	Name	txtBegin
Надпись (Label4)	Caption	Конец
Поле ввода (TextBox)	Name	txtEnd
Надпись (Label5)	Caption	Направление тура
Список (ListBox)	Name	lstTours
Календарь (Calendar)	Name	cldDate

15. В модуле формы наберите код. Для этого на свободном месте формы нажмите правую кнопку мыши и в появившемся меню выберите команду View Code.

```

Private M As Byte
Private Sub UserForm_Initialize()
lstTours.RowSource = "Туры!A1:A9"
txtBegin.Locked = True
txtEnd.Locked = True
End Sub
Private Sub cmdSave_Click()
Dim iCount As Integer
If bWin Then
iCount = Application.WorksheetFunction.CountA(Range("A:A")) + 1
Else
iCount = iFound
End If
Cells(iCount, 1).Value = txtLastName.Text
Cells(iCount, 2).Value = txtFirstName.Text
If optMale.Value Then Cells(iCount, 3).Value = "муж"
If optFemale.Value Then Cells(iCount, 3).Value = "жен"
Cells(iCount, 4).Value = lstTours.Text
Cells(iCount, 5).Value = txtBegin.Text
Cells(iCount, 6).Value = txtEnd.Text
End Sub
Private Sub cldDate_Click()
Select Case M
Case 1
txtBegin.Text = Format(cldDate.Value, "d/m/yyyy")
Case 2
txtEnd.Text = Format(cldDate.Value, "d/m/yyyy")
End Select
End Sub
Private Sub txtBegin_Enter()
M = 1
End Sub
Private Sub txtEnd_Enter()
M = 2
End Sub

```

```

Private Sub cmdCancel_Click()
Unload Me
End Sub
Private Sub cmdDelete_Click()
Rows(iFound).Delete
End Sub

```

16. Перейдем к конструированию формы для окна Поиск. В проекте добавьте форму (Insert - UserForm), на которой расположите две надписи (Label), поле ввода (TextBox), поле ввода со списком (ComboBox) и три кнопки (CommandButton) (см.рис.2)

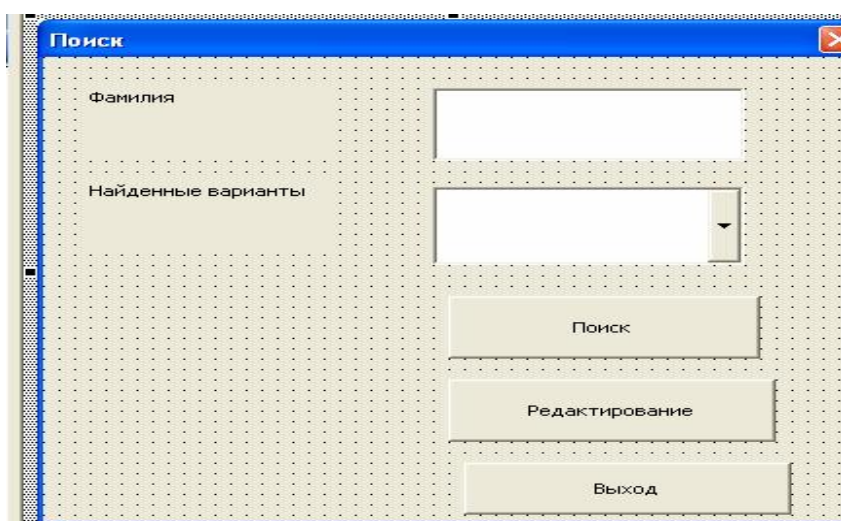


Рис.2. Форма окна Поиск.

17. При помощи окна Properties установите следующие значения свойств:

Форма	Name Caption	frmSearch Поиск
Кнопка (CommandButton)	Name Caption	cmdSearch Поиск
Кнопка (CommandButton)	Name Caption	cmdCancel ВЫХОД
Кнопка (CommandButton)	Name Caption	cmdEdit Редактирование
Надпись (Label)	Caption	Фамилия
Поле ввода (TextBox)	Name	txtLastName
Надпись (Label)	Caption	Найденные варианты
Поле со списком (ComboBox)	Name	cboFound

18. В модуле формы наберите код. Для этого на свободном месте формы нажмите правую кнопку мыши и в появившемся меню выберите команду View Code.

```

Private Sub UserForm_Initialize()
With cboFound

```

```

.Clear
.ColumnCount = 3
.ColumnWidths = "60;60;0"
End With
End Sub
Private Sub cmdSearch_Click()
Dim i As Integer, j As Integer, n As Integer, iRow As Integer
Dim sTest As String
cboFound.Clear
iRow = Application.CountA(Range("A:A"))
i = 2: j = 0
Do While i <= iRow
sTest = Cells(i, 1).Text
If IsNumeric(Application.Search(txtLastName.Text, sTest)) Then
cboFound.AddItem sTest
cboFound.List(j, 1) = Cells(i, 2).Text
cboFound.List(j, 2) = CStr(i)
j = j + 1
End If
i = i + 1
Loop
If cboFound.ListCount = 0 Then
MsgBox "Клиент не найден."
iFound = 0
Exit Sub
End If
End Sub
Private Sub cmdEdit_Click()
If cboFound.ListIndex = -1 Then
MsgBox "Сначала надо найти клиента"
Exit Sub
End If
iFound = cboFound.List(cboFound.ListIndex, 2)
Unload Me
bWin = False
With frmReg
.Caption = "Редактирование"
.cmdDelete.Visible = True
.txtLastName.Text = Cells(iFound, 1).Value
.txtFirstName.Text = Cells(iFound, 2).Value
If Cells(iFound, 3).Value = "муж" Then .optMale.Value = True
If Cells(iFound, 3).Value = "жен" Then .optFemale.Value = True
.txtBegin.Text = Cells(iFound, 5).Value
.txtEnd.Text = Cells(iFound, 6).Value
.Caption = "Редактирование"

```

```

.IstTours.MatchEntry = fmMatchEntryComplete
.IstTours.Text = Cells(iFound, 4).Value
.Show
End With
End Sub
Private Sub cmdCancel_Click()
Unload Me
End Sub

```

19. Проект готов. Начните заполнять нашу базу данных, используя кнопку Регистрация. Количество клиентов – 15 человек. Фамилии и все данные вводите произвольно.

Пример готовой базы данных Турфирма см.рис.3. Проверьте работоспособность кнопки

Поиск, при нажатии на нее появляется форма Поиск. Сохраните созданную работу в рабочей папке.

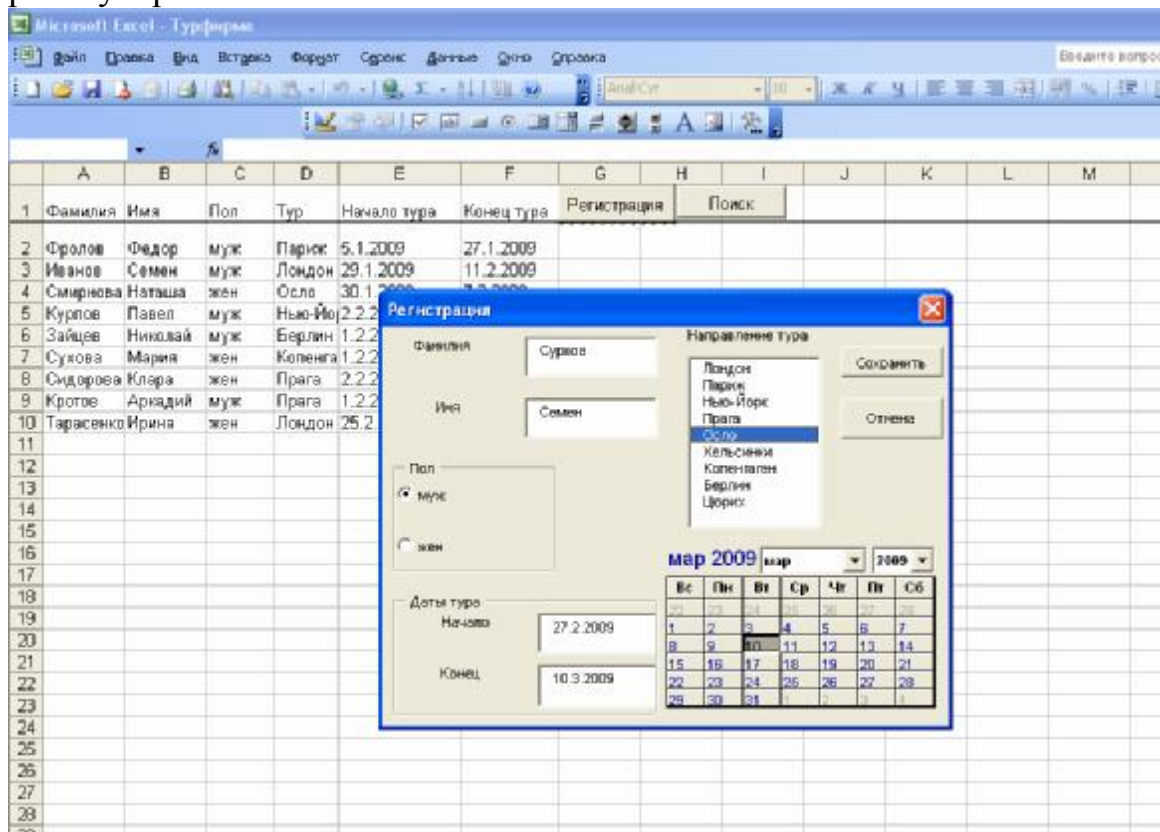


Рис.3. Готовая база данных Турфирма.

Лабораторная работа №13

Самостоятельная работа.

1. Создадим диалоговое окно, в котором находится сумма нечётных элементов матрицы либо найти р-элементов, находящихся на главной диагонали матрицы.
2. Запустите электронную таблицу Microsoft Excel.
3. Запустите редактор VBA. Выполните команду Сервис-Макрос-Редактор Visual Basic.
4. Добавьте форму. Выберите команду Insert-UserForm.
5. На ней расположите 3 ввода поля данных, 4 кнопки.
6. Создайте с помощью панели элементов 3 поля ввода данных (TextBox1,2,3), 4 командные кнопки (Command Button1,2,3,4).
7. При помощи окна Properties установите значение свойств:
 - Командная кнопка (Command Button1): Name – Command 1
Caption – Ввести построчно матрицу
 - Командная кнопка (Command Button4): Name – Command 2
Caption – Найти сумму нечётных элементов
 - Командная кнопка (Command Button3): Name – Command 3
Caption – Найти р-элементов, находящихся на главной диагонали
 - Командная кнопка (Command Button2): Name – Command 4
Caption – Выход
 - Поле ввода (Text Box1): Name – Text1
MultiLine – True
 - Поле ввода (Text Box2): Name – Text2
 - Поле ввода (Text Box3): Name – Text3
8. Создадим событийные процедуры для элементов формы:

```
Dim a( ) As Single
Dim n As Integer
Dim i As Integer
Dim g As Integer
Dim s As Single
Dim p As Integer
Private Sub Command1_Click()
n = Val(InputBox("Введите длину массива", "Матрица"))
Text1.Text = ""
ReDim a(n, n)
For i = 1 To n
For g = 1 To n
a(i, g) = Val(InputBox("Введите число", "Матрица"))
Text1 = Text1 + "" + Str(a(i, g))
```



```

Next g
Text1.Text = Text1.Text + Chr$(13)
Text1.Text = Text1.Text + Chr$(10)
Next i
End Sub
Private Sub Command3_Click()
s = 0
For i = 1 To n
For g = 1 To n
If (a(i, g) Mod 2 <> 0) Then s = s + a(i, g)
Next g
Next i
Text2 = "Сумма нечётных элементов=" + Str(s)
End Sub
Private Sub Command2_Click()
p = 1
For i = 1 To n
For g = 1 To n
If i = g Then p = p * a(i, g)
Next g
Next i
Text3 = "Произведение элементов на главной диагонали=" + Str(p)
End Sub
Private Sub Command4_Click()
End
End Sub

```

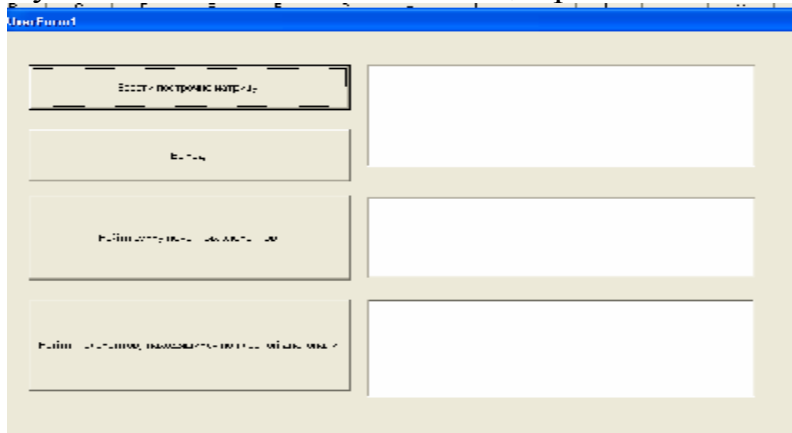
9. Проект готов. Нажмите клавишу F5, введите матрицу:

2 5 8

6 4 1

2 6 4

Сумма нечётных элементов = 6; Произведение элементов на главной = 32.

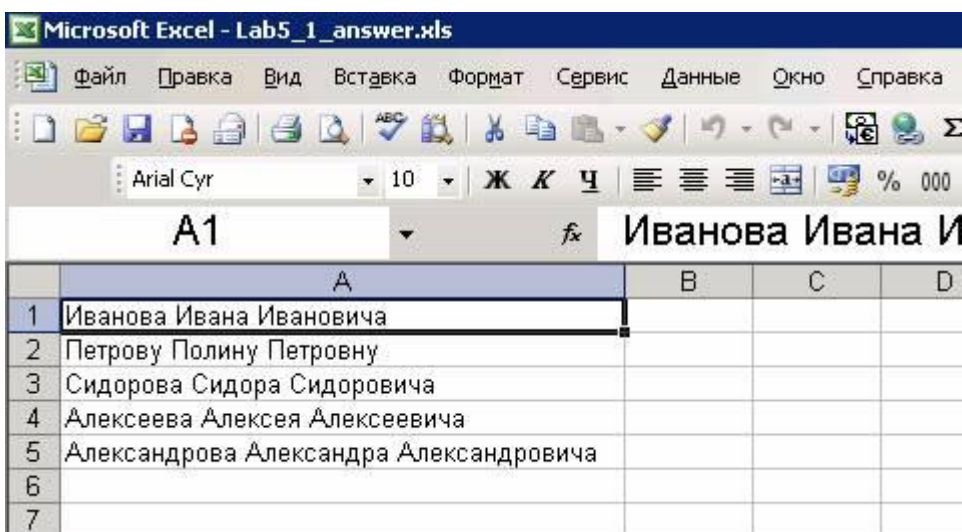


10. Сохраните созданную работу в рабочей папке.

Лабораторная работа №14

Работа с элементами управления

1. Создайте новую книгу Excel и сохраните ее как Prikaz.xls. Заполните ячейки с A1 по A5 значениями, аналогичными представленным на рис. 1.



The screenshot shows the Microsoft Excel interface with the following data in column A:

	A	B	C	D
1	Иванова Ивана Ивановича			
2	Петрову Полину Петровну			
3	Сидорова Сидора Сидоровича			
4	Алексеева Алексея Алексеевича			
5	Александрова Александра Александровича			
6				
7				

Рис.1. Список сотрудников на листе Excel

2. Откройте редактор Visual Basic и в окне Project Explorer щелкните правой кнопкой мыши по объекту "Эта книга" и в контекстном меню выберите **View Code**.

В окне редактора кода для этой книги введите следующий код:

'При открытии рабочей книги показываем форму UF1

```
Private Sub Workbook_Open()
```

```
UF1.Show
```

```
End Sub
```

'Специальная процедура, которая печатает приказ в Word

```
Public Sub DocWrite(sPovod As String, sFio As String, bFlagPremia As Boolean,  
bFlagGramota As Boolean, nSummaPremii As Long, sOtvIsp As String)
```

```
Set oWord = CreateObject("Word.Application")
```

```
Set oDoc = oWord.Documents.Add()
```

```
oWord.Visible = True
```

```
oDoc.Activate
```

```
With oWord.Selection
```

```
.TypeText "Приказ"
```

```
.Style = "Заголовок 1"
```

```
.ParagraphFormat.Alignment = wdAlignParagraphCenter
```

```
.TypeText vbCrLf
```

```
.Style = "Обычный"
```

```
.TypeText vbCrLf
```

```
.TypeText "г. Санкт-Петербург" & Space(90) & Date
```

```
.TypeText vbCrLf
```

```
.TypeText vbCrLf
```

```
.TypeText "За проявленные успехи в " & sPovod & " наградить " & sFio & ":"
```

```
.TypeText vbCrLf
```

```

If bFlagPremia Then
. TypeText vbTab & "- денежной премией в сумме " & nSummaPremii & "
рублей"
End If
If bFlagGramota Then
. TypeText ";"
. TypeText vbCrLf
. TypeText vbTab & "- почетной грамотой."
Else
. TypeText "."
End If
. TypeText vbCrLf
. TypeText vbCrLf
. TypeText vbCrLf
. TypeText vbCrLf
. TypeText "Генеральный директор" & vbTab & vbTab & vbTab & "Иванов
И.И."
. ParagraphFormat.Alignment = wdAlignParagraphCenter
. TypeParagraph
. TypeText vbCrLf
. TypeText vbCrLf
. ParagraphFormat.Alignment = wdAlignParagraphLeft
. TypeText Text:=("Отв. исполнитель" & sOtvIsp)
. TypeParagraph
End With
End Sub

```

3. Щелкните правой кнопкой мыши по вашему проекту Prikaz.xls и в контекстном меню выберите **Insert** -> **UserForm**. Выделите созданный вами объект формы и нажмите на кнопку <F4>. Настройте для свойства (**Name**) этой формы значение UF1.

4. Поместите на форму из **Toolbox** единственную кнопку — элемент управления **CommandButton1**. Установите для этой кнопки значение свойства **Caption** как "Напечатать приказ" и измените размеры и местонахождение этой кнопки, чтобы форма выглядела так, как представлено на рис. 2.

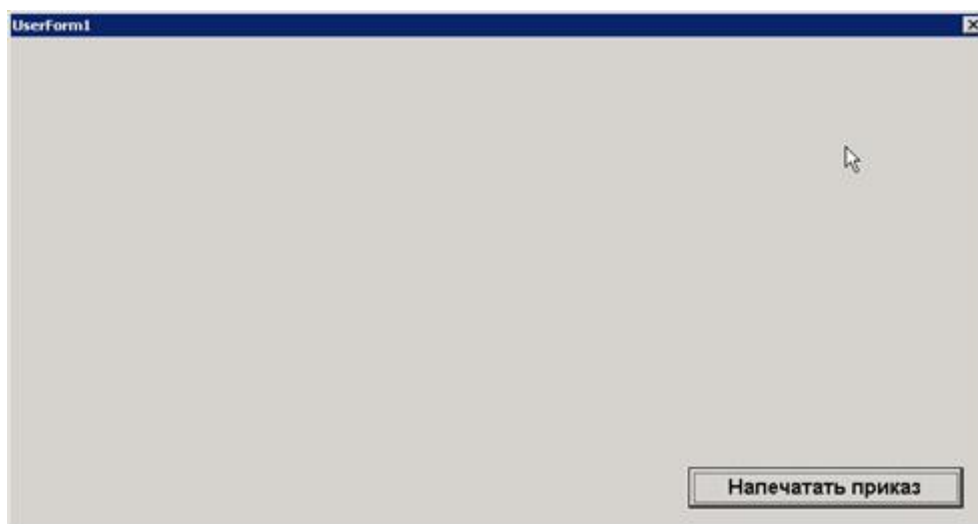


Рис.2. Форма — пока с единственной кнопкой

5. Щелкните правой кнопкой мыши по кнопке `CommandButton1` на вашей форме, в контекстном меню выберите **View Code** и добавьте в код событийной процедуры для события **Click** этой формы следующий код:

```
Private Sub CommandButton1_Click()
```

```
Dim sPovod As String
```

```
Dim sFio As String
```

```
Dim bFlagPremia As Boolean
```

```
Dim bFlagGramota As Boolean
```

```
Dim nSummaPremii As Long
```

```
Dim sOtvIsp As String
```

```
'Подставить данные из формы
```

```
sPovod = "освоении новых информационных технологий"
```

```
sFio = "Иванова Ивана Ивановича"
```

```
bFlagPremia = True
```

```
bFlagGramota = True
```

```
nSummaPremii = 100000
```

```
sOtvIsp = "Петрова П. П."
```

```
' Конец подстановки данных
```

```
Call ЭтаКнига.DocWrite(sPovod, sFio, bFlagPremia, bFlagGramota,  
nSummaPremii, sOtvIsp)
```

```
End Sub
```

6. Запустите вашу форму на выполнение и убедитесь, что она работает: выводит в создаваемый документ Word приказ с фиксированными значениями.

Задание:

Измените форму таким образом, чтобы вместо присвоения переменным в выделенном комментариями блоке заранее определенных значений пользователь смог выбирать данные при помощи формы. При этом:

- Значение переменной sPovod должно выбираться из трех возможных значений: "освоении новых информационных технологий", "внедрении новых программных продуктов" и значение, которое пользователь может ввести через текстовое поле. Используйте для этого набор из трех переключателей и текстовое поле (оно должно быть скрыто, если пользователь выбрал один из двух других переключателей). По умолчанию должно подставляться "освоении новых информационных технологий".
- Значение переменной sFio должно выбираться пользователем при помощи комбинированного списка. В этот комбинированный список должны автоматически помещаться значения из всех непустых ячеек столбца А листа Excel. По умолчанию должно выбираться значение "Иванова Ивана Ивановича".

Примечание.

Образец для работы с ячейками столбца можно получить из ответов к предыдущим лабораторным работам.

- Значения переменных bFlagPremia и bFlagGramota должны устанавливаться в зависимости от состояния двух флажков — "Премия" и "Грамота". По умолчанию оба флажка должны быть установлены.
- Если пользователь снял оба флажка, то ему должно выводиться предупреждающее сообщение "Не выбрана ни премия, ни почетная грамота!" с отменой вывода документа.
- Пользователь должен иметь возможность задавать значение переменной nSummaPremii либо при помощи полосы прокрутки с диапазоном значений от 0 руб. до 100 000 руб., либо при помощи текстового поля. Если флажок "Премия" снят, то полоса прокрутки и текстовое поле должны быть скрыты от пользователя.
- Ход полосы прокрутки должен быть равен 100 рублям.
- По умолчанию размер премии должен быть равен 100 рублям.
- Поместите на форму еще одну кнопку "Отмена". Эта кнопка должна закрывать текущую форму и срабатывать при нажатии на клавишу <Esc>.
- В заголовке формы должно выводиться значение "Формирование приказа о выплате премии".

Общий вид формы может выглядеть, например, так, как представлено на рис.3:

Рис. 3. Готовая форма

Лабораторная работа №15

Самостоятельная работа

1. Вы достаточно потрудились, самое время отдохнуть. Выполним работу, которая отвлечет Вас. Запустите электронную таблицу Microsoft Excel.
2. Запустите редактор VBA. Выполните команду Сервис – Макрос – Редактор VB.
3. В окне Project- VBAProject для ЭтаКнига правой кнопкой мыши выполните View – Code.
4. В появившемся окне введите код:

```

Sub ShowStars()
Randomize 'Генератор случайных чисел
StarWidth = 100 'Ширина звезды
StarHeight = 100 'Высота звезды
For i = 1 To 10 'Цикл = количество звезд
TopPos = Rnd() * (ActiveWindow.UsableHeight - StarHeight) 'Позиция по
вертикали
LeftPos = Rnd() * (ActiveWindow.UsableWidth - StarWidth) 'Позиция по
горизонтали
Set NewStar = ActiveSheet.Shapes.AddShape _
(msoShape4pointStar, LeftPos, TopPos, StarWidth, StarHeight)
'Непосредственно добавление звезды
NewStar.Fill.ForeColor.SchemeColor = Int(Rnd() * 56) 'Заливка случайным
цветом
Application.Wait Now + TimeValue("00:00:01") 'Ждем прорисовки для эффекта
появления
DoEvents 'Даем обновиться данным

```

```
Next i
Application.Wait Now + TimeValue("00:00:02") 'Пауза перед тем как убирать
звезды
Set myShapes = Worksheets(1).Shapes 'Все рисованные объекты. В нашем
случае - звезды нарисованные нами
For Each shp In myShapes 'перебираем в цикле все звезды
If Left(shp.Name, 9) = "AutoShape" Then 'Если это наши объекты, то
shp.Delete 'удаляем их
Application.Wait Now + TimeValue("00:00:01") 'Ждем для эффекта угасания
End If
Next
Worksheets(1).Shapes("Message").Visible = True
End Sub
```

5. Запустите программу, нажав клавишу F5, и переключитесь с панели задач на Excel.

Наслаждаемся проделанной работой.

6. Сохраните работу в рабочей папке.

Список литературы

1. Гарнаев А. Excel, VBA, Internet в экономике и финансах. Санкт-Петербург, 2005.
2. Программирование в пакетах MS OFFICE. Под редакцией профессора С.В.Назарова. М: Финансы и статистика, 2007.
3. Слепцова Л.Д. Программирование на языке VBA. Самоучитель, - М.: Издательский дом «Вильямс», 2004
4. Браун С. Visual Basic 5 с самого начала – СПб : Питер, 1998

ИНФОРМАТИКА
VBA в Office

Елена Владимировна Ефимова

Ответственная за выпуск
директор издательства

Изд. №	Подписано к печати	Объем 3 уч.-изд.л.
Печать офсетная	Печать офсетная	Гарнитура
«Таймс»		
Формат 60x84/16	Тираж 200 экз.	Заказ №

344002, г. Ростов-на-Дону, ул. Б.Садовая, 69,РГЭУ «РИНХ». Издательство.
Отпечатано