Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра экономической математики, информатики и статистики (ЭМИС)

ИСПОЛЬЗОВАНИЕ УПРАВЛЯЮЩИХ ЭЛЕМЕНТОВ (ПАНЕЛЬ ЭЛЕМЕНТОВ VISUAL BASIC)

Отчет по практической работе по дисциплине «Информационные технологии»

Студент группы 549
 Баулин С.К.
«__» _____ 2020 г.
Старший преподаватель
кафедры ЭМИС
_____ _____ Афанасьева И. Г.
«__» _____ 2020 г.

Томск 2020

Практическая работа №9

Использование управляющих элементов

(панель элементов Visual Basic)

Цель работы: знакомство с управляющими элементами пользовательской формы.

Теоретический материал

Панель элементов представляет собой окно, внутри которого находятся значки различных элементов, используемых в приложениях.

Чтобы работать с элементами в приложениях, программист должен:

- понимать, что такое свойства, события и методы соответствующего элемента;
- уметь использовать свойства, события и методы элемента.

Свойства – атрибуты объекта, которые изменяют внешний вид объекта и его поведение.

События – действие, распознаваемое объектом, для которого можно запрограммировать отклик.

Метод – команда, которую Вы отдаете объекту. При помощи методов можно приказать объекту выполнить те или иные действия, например, заставить выгрузиться из памяти форму.

Рассмотрим наиболее часто используемые элементы.

Форма – это визуальная основа приложений Visual Basic (рисунок 1). Любое приложение, выводящее информацию на экран, построено на основе формы того или иного типа.

Чтобы создать новую форму, выполните команду **Insert – UserForm**. На экране появится форма, состоящая из нескольких компонентов.

Граница формы придает необходимую степень гибкости. Все эти возможности задаются при помощи свойств **BorderStyle**.

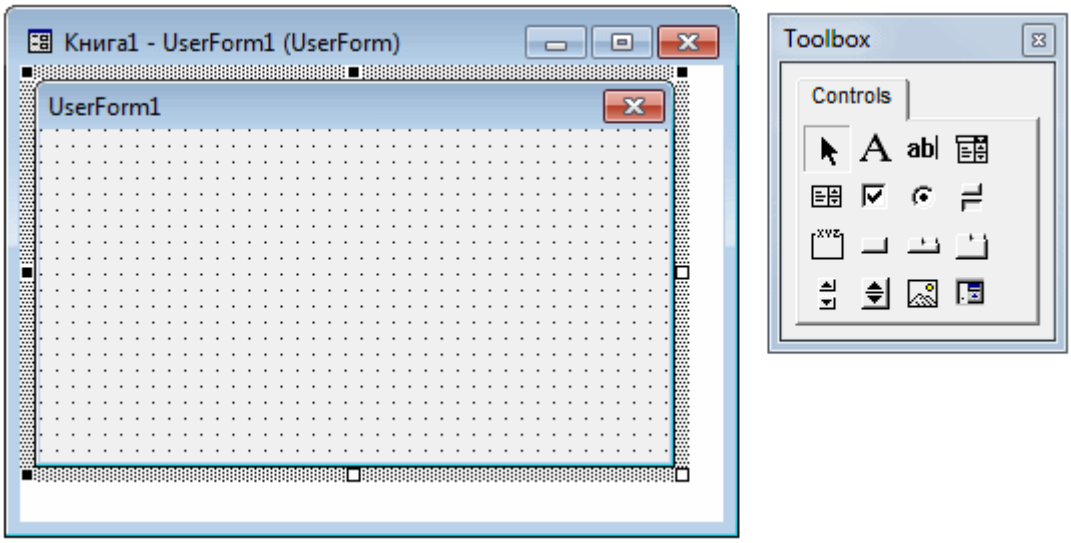

Рисунок 1 – Скриншот объекта UserForm

Название – текст, выводимый в заголовке формы. В нем может содержаться имя приложения, краткое описание формы или информация о текущем состоянии.

Чтобы изменить название формы, следует присвоить нужный текст свойству **Caption** в окне свойств.

Свойства формы

Свойство **BackColor** определяет цвет фона для формы.

Свойство **BorderStyle** определяет особенности границы, окружающей форму.

Значение	Описание
0 – None	Позволяет оформлять вид без границ
1 – Fixed Single	Позволяет оформлять вид одинарной границей

Свойство **Caption** определяет текст, который выводится в заголовке формы.

Свойство **ForeColor** определяет цвет текста, выводимого на форме.

Свойство **Height** определяет высоту формы.

Свойство **Width** определяет ширину формы.

Свойство **Left** определяет расстояние формы от левого края экрана.

Свойство **Top** определяет расстояние формы от верхнего края экрана

Данные значения задаются в твипах (twips). Соотношение размеров твипа и пиксела изменяется в зависимости от разрешения экрана, и измеряется при помощи переменной Screen.TwipsPerPixelX и Screen.TwipsPerPixelY для горизонтальных и вертикальных размеров соответственно.

Свойство **Name** определяет имя объекта, используемое в дальнейшем процедурой.

События форм

Событие **Activate** активизирует форму. Активизация формы производится после ее инициализации. С событием Activate тесно связаны события Initialize, Load, GotFocus.

Между этими событиями существуют достаточно тонкие отличия, причем главное из них – порядок, в котором события возникают в приложении. Этот порядок выглядит так:

- **Initialize**. Событие происходит во время конфигурации и до загрузки формы.
- **Load**. Событие происходит после инициализации формы, но до ее отображения на экране. Добавляя код в процедуру события Load, Вы можете настроить внешний вид или поведение формы.
- **Activate**. Событие происходит после загрузки формы в память, но до того, как форма станет активной.
- **GotFocus**. Это событие, если оно происходит, возникает при получении фокуса формой – когда форма загружается или когда пользователь обращается к ней, щелкая мышью.

После открытия формы из перечисленных событий могут произойти только GotFocus или Activate – хотя в отдельных, очень специфических случаях может произойти и событие Initialize.

Событие Initialize происходит, когда Visual Basic впервые узнает о существовании формы. В режиме выполнения это происходит сразу же после команды **Run – Start**. За ним следует событие Load – оно происходит, когда Visual Basic загружает форму. После загрузки формы и передачи ей фокуса

(другими словами, при активизации формы) происходит событие Activate. Через считанные миллисекунды после него происходит событие GotFocus. Тем не менее, событие GotFocus может произойти лишь в том случае, если на форме нет ни одного элемента. Если же на форме есть видимый элемент, то фокус получает он, а событие GotFocus формы будет пропущено – вместо него будет вызвано событие GotFocus элемента.

Следовательно, в нормальной ситуации при запуске приложения будет вызвано событие Load первой отображаемой формы, за которым последует событие Activate. Разумеется, приложение может иметь и другие окна. Когда пользователь или программа повторно переключится в первое окно, снова будет вызвано событие Activate, но на этот раз без Load. Впрочем, событие Load может быть повторно вызвано, если форма была выгружена во время выполнения программы.

Событие **Deactivate** по смыслу противоположно Activate. Оно происходит в том случае, если форма перестает быть активной, т.е., когда фокус передается другой форме или приложению. В зависимости от выбранной цветовой схемы Windows цвет заголовка формы может измениться.

Событие **Unload** по смыслу противоположно Load. Чаще всего процедура этого события используется для того, чтобы спросить у пользователя, действительно ли он желает закрыть форму. Если Вы посмотрите на процедуру Unload в окне программы, то увидите, что она немного отличается от процедур других событий. За именем процедуры следует аргумент (Cancel As Integer), с его помощью можно отменить выгрузку формы.

Событие **Resize** происходит, когда пользователь изменяет размеры формы. Чаще всего оно применяется в двух случаях:

- для масштабирования управляющих элементов, размещенных на форме;
- для восстановления исходных размеров формы.

В обоих случаях используются свойства Height и Width.

Кнопки (CommandButton) являются самым распространенным элементом любого приложения.

Свойство	Описание
Cancel	Если данное свойство имеет значение True, нажатие клавиши Esc приводит к закрытию формы. Если на форме несколько кнопок, то лишь у одной кнопки свойство Cancel может иметь значение True
Caption	Определяет текст, который выводится на кнопке.
Default	Если данному свойству задать значение True, пользователь сможет нажать кнопку при помощи клавиши Enter. Если на форме несколько кнопок, то лишь у одной кнопки свойство Default может иметь значение True
Enable	Если установить для данного свойства значение False, кнопка станет недоступной. Кнопка остается на экране
Name	Имя кнопки, по которому в программе на Visual Basic отличается одна кнопка от других
Picture	Если вы хотите, чтобы в нормальном, не нажатом состоянии на кнопке присутствовал рисунок, задайте имя графического файла в данном свойстве
Style	Выбирая значения данного свойства можно оставить кнопку чисто текстовой или поместить на нее рисунок
TabIndex	Определяет порядок перебора элементов клавишей Tab. Если данное свойство определено значением 0, то соответствующий элемент получает фокус первым (при условии, что он видим и не заблокирован). При изменении значения данного свойства у одного элемента, изменяется порядок перебора и у других элементов
TabStop	Если определить данное свойство значением False, пользователь не сможет перейти к соответствующему элементу клавишей Tab. Но мышью элемент активизируется (при условии, что он видим и не заблокирован)
Visible	Если установить для данного свойства значение False, кнопка станет недоступной. Кнопка исчезает с экрана

События кнопок рассмотрим на примере события Click – реакция кнопки на нажатие.

```
Private Sub CommandButton1_Click()
UserForm1.Hide
End Sub
```

Даная процедура позволяет при нажатии на кнопку скрывать пользовательскую форму.

Методы кнопок рассмотрим на примере метода SetFocus, который используется для передачи фокуса конкретной кнопке.

```
Private Sub UserForm_Initialize()
```

CommandButton1.SetFocus

End Sub

Данная процедура позволяет сделать активной кнопку CommandButton при инициализации пользовательской формы UserForm.

Текстовые поля (TextBox) обычно применяются для ввода данных или для получения информации от пользователя.

Свойство	Описание
Locked	Если данное свойство определить значением True, то текстовое поле будет доступно только для вывода информации
MaxLength	Ограничивает длину вводимого текста заданным количеством символов
MultiLine	Позволяет вывести текст в несколько строк
Name	Программное имя поля. Рекомендуется использование префикса имени txt
PasswordChar	Задает символ, вводимый в данное поле (наиболее распространен символ *)
ScrollBar	Позволяет выполнять прокрутку информации в поле
SelLength	Количество символов, которые будут выделены. Доступно только в программном коде
SelStart	Порядковый номер символа, после которого будет мигать курсор. Доступно только в программном коде
SelText	Автоматически заменяет выделенный текст. SelText=«NewText». Доступно только в программном коде
TabIndex	Определяет порядок перебора текстовых полей формы
Text	Содержимое поля

События текстового поля рассмотрим на примере события Change. Оно происходит каждый раз, когда пользователь вставляет, заменяет или удаляет символы текстового поля.

Private Sub TextBox1_Change()

TextBox1.Text = "s"

End Sub

В данной процедуре при изменениях в текстовом поле TextBox1 будет отображаться символ S.

Методы текстовых полей рассмотрим на примере метода SetFocus – передать фокус текстовому полю.

Private Sub UserForm_Initialize()

TextBox1.SetFocus

End Sub

Надписи (Label) используется для вывода текста, но пользователь не может по своему усмотрению изменить текст надписи.

Свойство	Описание
BackColor	Цвет фона для поля надписи
BorderStyle	Используется совместно с BackColor. Если назначить данному свойству значение = 1, а BackColor определить белым цветом, то надпись будет выглядеть как текстовое поле, но останется доступной только для чтения
Caption	Определяет текст, который содержится в надписи
Name	Определяет программное имя надписи. Рекомендуется префикс имени lbl
TabIndex	Определяет порядок перебора элементов
UseMnemonic	Если необходимо в тексте надписи прописать символ &, то данному свойству определяют значение False
WordWrap	Позволяет определить динамическое изменение размеров надписи при изменении ее содержимого

Переключатели (OptionButton) позволяют выбрать один вариант из группы. Обычно переключатели группируются в рамках (Frame).

Свойство	Описание
Caption	Текст, выводимый на поле переключателя
Name	Программное имя переключателя
Value	Если установлено значение True в режиме конструирования, то данный переключатель активен после запуска программы

В **событиях** необходимо обратить внимание на событие Click.

Список (ListBox) применяется для хранения списка значений. Из списка пользователь может выбрать одно или несколько значений, которые впоследствии будут использоваться в тексте программы.

Основные свойства

Enabled	Допустимые значения: True (запрещен выбор значения из списка пользователем) и False (в противном случае)
Text	Возвращает выбранный в списке элемент
List	Возвращает элемент списка, стоящий на пересечении указанных строки и столбца. List (row, column)
MultiSelect	Устанавливает способ выбора элементов списка. Допустимые значения: fmMultiSelectSingle (выбор только одного элемента) fmMultiSelectMulti (разрешен выбор нескольких элементов посредством либо щелчка, либо нажатием клавиши <Пробел>) fmMultiSelectExtended (разрешено использование клавиши <Shift> при выборе ряда последовательных элементов списка)
Selected	Используется для определения выделенного текста, когда свойство MultiSelect имеет значение fmMultiSelectMulti или fmMultiSelectExtended. Допустимые значения False, True
ColumnCount	Устанавливает число столбцов в списке
ListCount	Возвращает число элементов списка
ListIndex	Возвращает номер текущего элемента списка. Нумерация элементов начинается с 0

Методы элемента ListBox

Clear	Удаляет все элементы из списка
RemoveItem	Удаляет из списка элемент с указанным номером RemoveItem (index)
AddItem	Добавляет элемент в список AddItem ([item [, varIndex]]) Item – элемент (строковое выражение), добавляемое в список varIndex – номер добавляемого элемента

Заполнение списка

Заполнить список можно одним из следующих способов.

Поэлементно (список состоит из одной колонки)	ListBox1.AddItem "Hello" ListBox1.AddItem "Bye" ListBox1.ListIndex=0
Массивом, если список состоит из одной колонки	ListBox1.List=Array ("Hello", "Bye") ListBox1.ListIndex=1
Поэлементно, если список состоит из нескольких колонок	ListBox1.ColumnCount =2 ListBox1.AddItem "Hello" ListBox1.List(0,1)= "Gane" ListBox1.AddItem "Hello" ListBox1.List(1,1)= "Lisa" ListBox1.AddItem "Hello" ListBox1.List(2,1)= "Dim"
Массивом, если список состоит из нескольких колонок	Dim A(2,1) As String A(0,0)= "Hello" A(0,1)= "Gane" A(1,0)= "Hello" A(1,1)= "Lisa" A(2,0)= "Hello" A(2,1)= "Dim" ListBox1.ColumnCount=2 ListBox1.List=A

Выбор нескольких элементов из списка

Вычисление среднего значения выбранных в списке элементов:

```
ListBox1.List = Array (1,2,3,4,5,6,7,8)
ListBox1.ListIndex = 0
ListBox1.MultiSelect = fmMultiSelectMulti
Среднее = 0
N = 0
For i = 0 to ListBox1.ListCount - 1
  If ListBox1.Selected(i) Then
    N = N + 1
    Среднее = Среднее + ListBox1.List(i)
  End if
Next
Среднее = Среднее / N
```

Переключатель.Флажок (CheckBox)

Элемент управления (OptionButton) позволяет выбрать из нескольких взаимоисключающих параметров или действий. Переключатели типа

CheckBox обычно изображаются группами, обеспечивая возможность выбора альтернативного варианта.

Value	Возвращает True, если переключатель выбран и False в противном случае
Enabled	True – пользователь может выбрать переключатель, False – в противном случае
Visible	True – переключатель отображается во время выполнения программы, False – в противном случае
Capture	Надпись, отображаемая рядом с переключателем

Комбинированные поля (Combobox) сочетают возможности текстового поля и списка. Комбинированное поле позволяет выбрать из списка заранее определенную строку или ввести значение, которого нет в списке. Существует три разновидности комбинированных полей, выбираемые в режиме конструирования: раскрывающиеся комбинированные поля, простые комбинированные поля и раскрывающиеся списки.

По свойствам, событиям и методам комбинированные поля очень похожи на списки. Тем не менее, свойство Text в комбинированных полях работает несколько иначе. Если для списков свойство Text при выполнении программы может лишь вернуть текст текущей выделенной строки, то для комбинированных полей значение этого свойства можно задавать и во время выполнения – текст задается даже в том случае, если строка отсутствует в списке. Особую роль для комбинированного поля играет свойство Style. Оно может принимать три значения, определяющих поведение и внешний вид комбинированного поля.

0 - Раскрывающееся комбинированное поле похоже на стандартное текстовое поле, справа от которого имеется кнопка со стрелкой. Если нажать кнопку, под текстовым полем раскрывается список. Пользователь может либо выбрать строку из списка, либо внести в поле свой собственный вариант. Именно этот вариант обычно называется комбинированным полем.

1 - Простое комбинированное поле представляет собой разновидность описанного выше – единственное отличие состоит в том, что

список постоянно остается открытым. Этот вариант выбирается в том случае, если на Вашей форме остается много свободного места.

События комбинированного поля.

Событие	Описание
Click	Реакция на щелчок мыши
DblClick	Реакция на двойной щелчок мыши. Имеет значение лишь для простых полей

Методы комбинированных полей совпадают с методами списков.

Ход выполнения работы

Задание 1.

1. Вставьте новую форму. Высота формы – 150, ширина – 200.
2. Поместите на ней два объекта CommandButton. Одну кнопку назвать «Уменьшить», вторую – «Увеличить».
3. При нажатии на кнопку «Увеличить» размер формы должен увеличиваться на 10, не превышая максимальных значений: для высоты – 450; для ширины – 600.
4. При нажатии на кнопку «Уменьшить» размер формы должен уменьшаться на 10, не переходя за минимальный размер формы.

На рисунках 2 – 4 изображено выполненное задание №1.

```
Private Sub large_button_Click()

If Form1.Width <= 600 And Form1.Height <= 450 Then
   Form1.Width = Form1.Width + 10
   Form1.Height = Form1.Height + 10
End If
Label1.Caption = "Высота формы: " & Form1.Height & "   Ширина формы: " & Form1.Width

End Sub
```

```
Private Sub small_button_Click()
If Form1.Width > 200 And Form1.Height > 150 Then
   Form1.Width = Form1.Width - 10
   Form1.Height = Form1.Height - 10
End If
Label1.Caption = "Высота формы: " & Form1.Height & "   Ширина формы: " & Form1.Width
End Sub
```

Рисунок 2 – Скриншот кода формы для задания 1

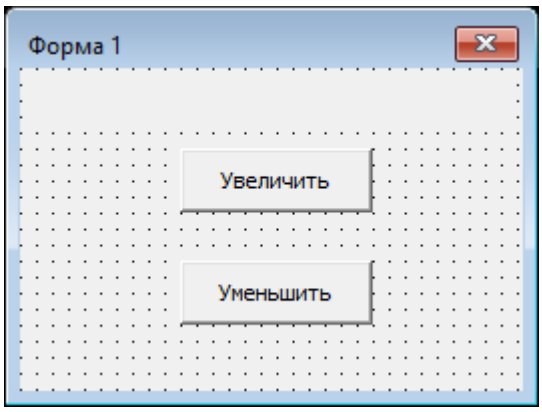

Рисунок 3 – Скриншот формы для задания 1

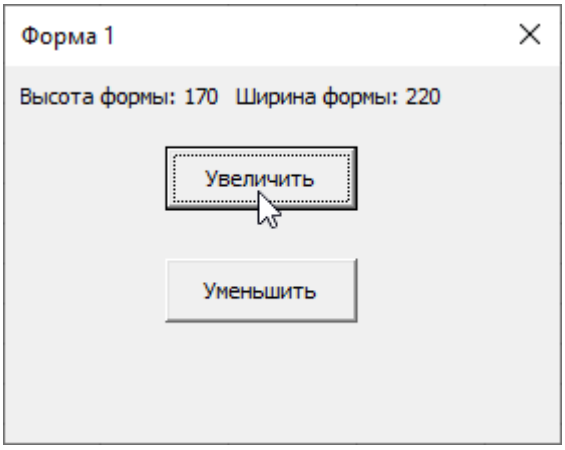

Рисунок 4 – Скриншот результата работы формы для задания 1

Задание 2.

1. Откройте форму предыдущего задания.
2. Дважды щелкните по форме, чтобы перейти к окну программы. В двух раскрывающихся списках, раскрывающихся в верхней части окна программы, должны быть выбраны строки Form и Load; это означает, что в настоящий момент Вы работаете с событием Load формы.
3. Раскройте второй список и прокручивайте его вниз то тех пор, пока не найдете в нем строку Resize. Тем самым Вы переходите к событию Resize данной формы.
4. Введите в событие Resize следующую строку:

Width = Height

5. Запустите проект.
6. Попробуйте изменить размеры формы различными способами. Как ведет себя форма? - Ширина и высота формы становится одинаковой.
7. Остановите программу.

На рисунках 5 – 7 изображено выполненное задание №2.

```
Private Sub UserForm_Resize()
Width = Height
End Sub
```

Рисунок 5 – Скриншот кода формы для задания 2

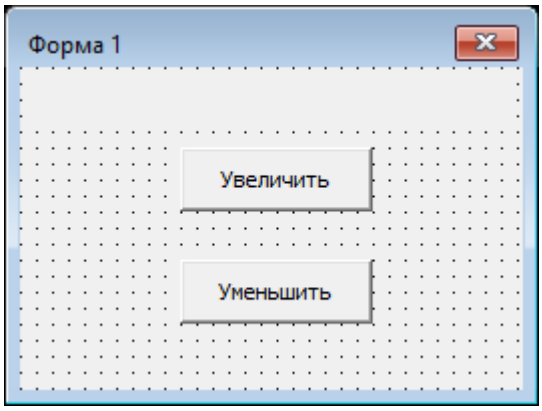

Рисунок 6 – Скриншот формы для задания 2

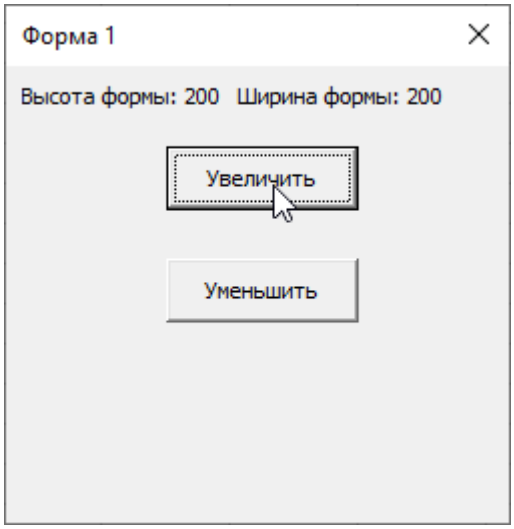

Рисунок 7 – Скриншот результата работы формы для задания 2

Задание 3.

Создать форму, которая позволяет выбрать несколько чисел, выводимых в списке, заполняемом случайными значениями при инициализации окна, в диалоговом окне **Операции над элементами** (рисунок 8).

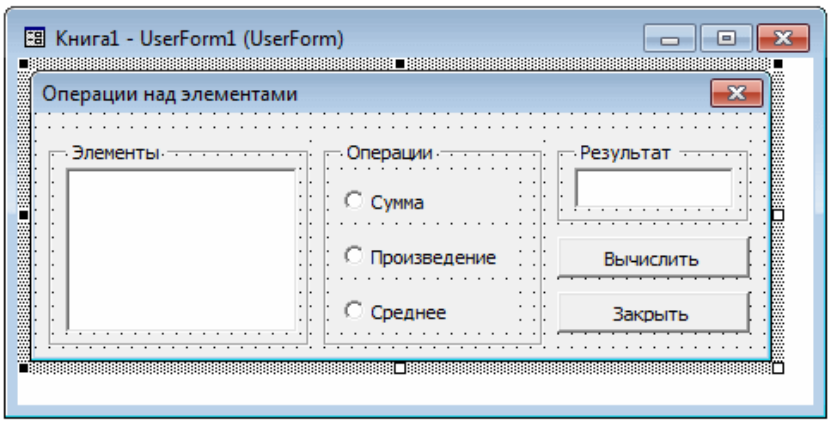

Рисунок 8 – Скриншот объекта UserForm

В группе Операции следует установить один из переключателей: **Сумма, Произведение** или **Среднее**, чтобы указать какая из операций будет выполняться над выбранными числами. Нажатие кнопки **Вычислить** должно привести к выполнению операции и выводу результата в поле **Результат**. Кнопка **Закрыть** выполняет закрытие диалогового окна, а также должен быть запрет для ввода в поле **Результат** объекта UserForm.

На рисунках 9 – 14 изображено выполненное задание №3.

```
Dim iter As Integer

Private Sub close_button_Click()
Unload Form2
End Sub

Private Sub count_button_Click()
Dim Sum, Mult, N As Integer

N = 0
Mult = 1
AverC = 0

If sum_check.Value Then
For i = 0 To elements.ListCount - 1
 If elements.Selected(i) Then
    Sum = Sum + elements.List(i)
End If
Next
result_text.Text = Sum

End If

If mult_check.Value Then
For i = 0 To elements.ListCount - 1
 If elements.Selected(i) Then
    Mult = Mult * elements.List(i)
End If
Next
result_text.Text = Mult
End If

If average_check.Value Then
For i = 0 To elements.ListCount - 1
 If elements.Selected(i) Then
    N = N + 1
    AverC = AverC + elements.List(i)
End If
Next
result_text.Text = AverC / N
End If


ActiveWorkbook.Sheets("Результаты").Range("A" & iter).Value = result_text.Value
iter = iter + 1

End Sub
```

Рисунок 9 – Скриншот кода формы для задания 3 (1)

```
Private Sub UserForm_Initialize()
iter = 1
   elements.MultiSelect = fmMultiSelectMulti
   Dim ListItem(10) As Integer

   DirectionP = "C:\Users\seron\Desktop\study\3sem"
   Workbooks.Open(DirectionP & "\Данные.xlsx").Activate

   If ActiveWorkbook.Saved Then
   For i = 0 To 10
      ListItem(i) = -20 * Rnd + 10
      ListItem(i) = ActiveWorkbook.Sheets("Данные").Range("A" & i + 1).Value
   Next

  ListItem = Sheets("Данные").Range("A1:A10").Value
  elements.List = ListItem

   'elements.RowSource = "Данные!$A$1:$A$10"

   Else
   MsgBox "Книга с данными не была сохранена"
   End If

End Sub
```

Рисунок 10 – Скриншот кода формы для задания 3 (2)

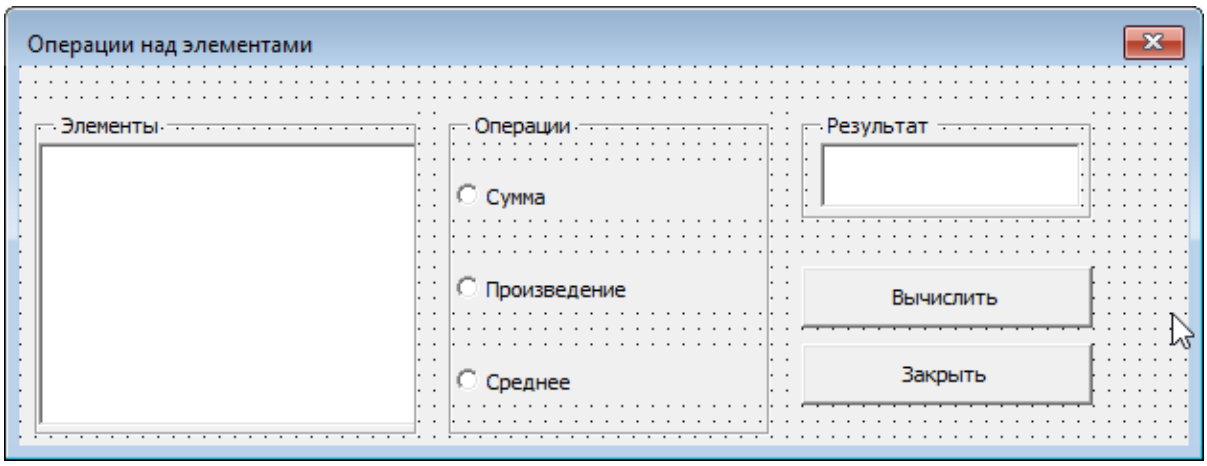

Рисунок 11 – Скриншот формы для задания 3

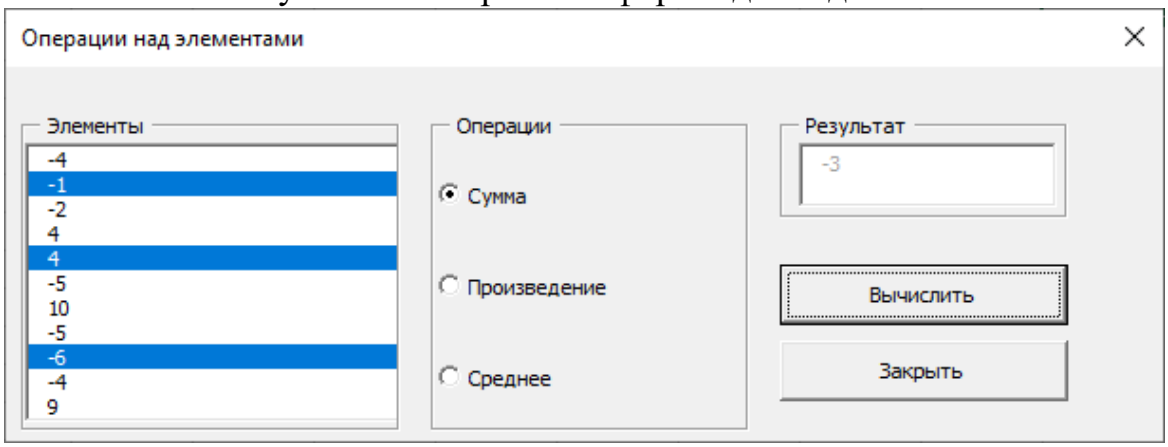

Рисунок 12 – Скриншот результата работы формы для задания 3 (1)

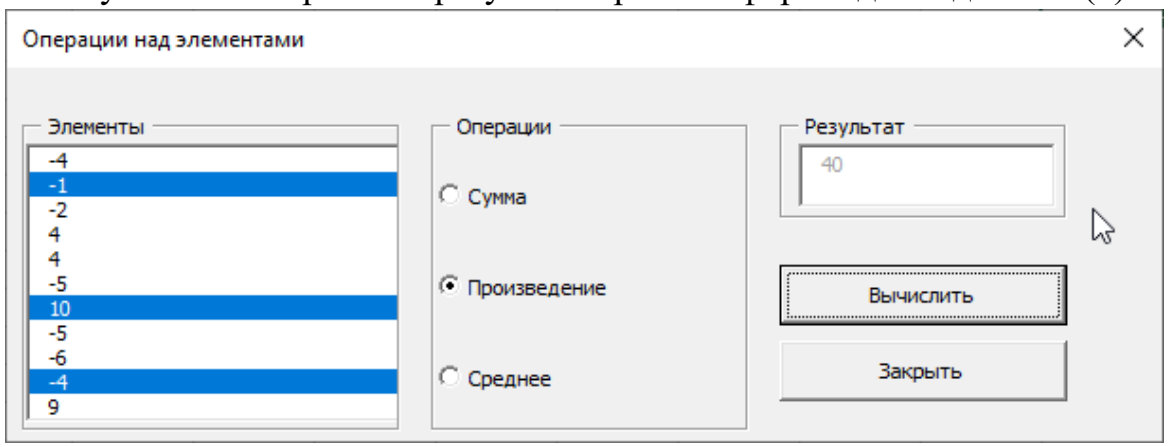

Рисунок 13 – Скриншот результата работы формы для задания 3 (2)

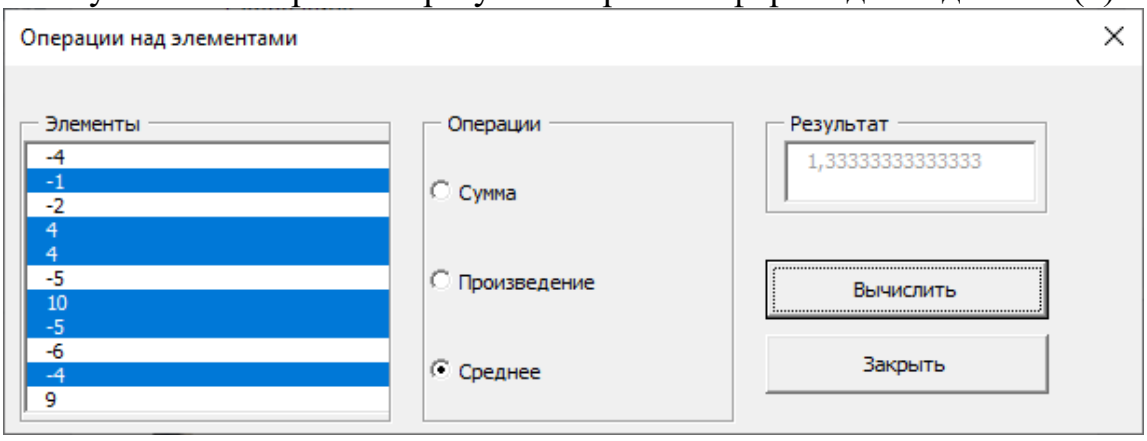

Рисунок 14 – Скриншот результата работы формы для задания 3 (3)

Вывод по лабораторной работе: в ходе лабораторной работы было произведено знакомство с управляющими элементами пользовательской формы VBA.