

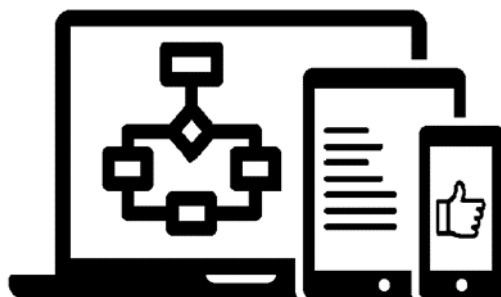
МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Программное обеспечение информационных технологий»

ИНФОРМАТИКА

*Методические рекомендации к лабораторным работам
для студентов специальности 6-05-0715-07 «Эксплуатация
наземных транспортных и технологических машин
и комплексов» дневной формы обучения*

Часть 2



Могилев 2024

УДК 004
ББК 32.973
И74

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Программное обеспечение информационных технологий» «30» января 2024 г., протокол № 7

Составитель ст. преподаватель О. А. Пономарева

Рецензент канд. техн. наук, доц. В. М. Ковальчук

Методические рекомендации к лабораторным работам предназначены для студентов специальности 6-05-0715-07 «Эксплуатация наземных транспортных и технологических машин и комплексов» дневной формы обучения.

Учебное издание

ИНФОРМАТИКА

Часть 2

Ответственный за выпуск

В. В. Кутузов

Корректор

И. В. Голубцова

Компьютерная верстка

Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 16 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».

Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.

Пр-т Мира, 43, 212022, г. Могилев.

© Белорусско-Российский
университет, 2024

Содержание

1 Лабораторная работа № 12. Программирование на алгоритмическом языке	4
2 Лабораторная работа № 13. Программирование на алгоритмическом языке	16
3 Лабораторная работа № 14. Программирование на алгоритмическом языке	21
4 Лабораторная работа № 15. Программирование на алгоритмическом языке	27
5 Лабораторная работа № 16. Программирование на алгоритмическом языке	32
Список литературы	38

1 Лабораторная работа № 12. Программирование на алгоритмическом языке

Цель работы: изучить возможности создания и использования пользовательских форм; изучить процедуры обработки ошибок и использование их при решении задач.

1.1 Теоретические сведения

В VBA можно использовать пользовательские (настраиваемые) диалоговые окна в создаваемых программах при помощи добавления в проект объекта *UserForm*. Пользовательская форма представляет собой пустое диалоговое окно (рисунок 1.1), на которое, в зависимости от решаемой задачи, размещаются нужные элементы управления, используя панель инструментов *Toolbox*.

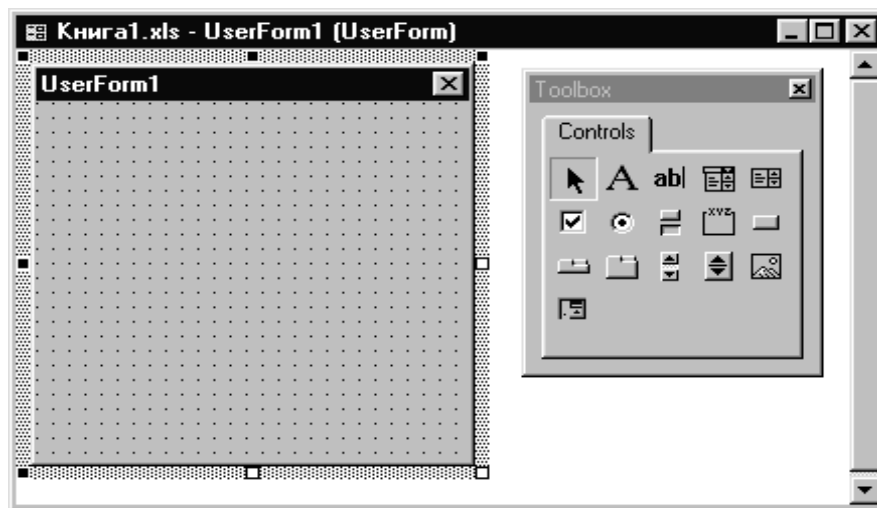


Рисунок 1.1 – Окно редактирования форм и панель инструментов

Размеры формы и расположенных на ней элементов управления можно изменять. Технология изменения размеров стандартная для Windows: выделить изменяемый элемент, разместить указатель мыши на одном из размерных маркеров и протащить его при нажатой левой кнопки мыши так, чтобы объект принял требуемые размеры. Окно редактирования форм поддерживает операции буфера обмена, т. е. можно копировать, вырезать и вставлять элементы управления, расположенные на поверхности формы. Для облегчения размещения и выравнивания элементов управления используется сетка. Список основных элементов управления, их назначение и соответствующие кнопки панели элементов приведены в таблице 1.1.

После размещения элементов управления на форме необходимо связать объект на форме с кодом. Для выполнения данной операции следует дважды щелкнуть по элементу управления в форме, появится окно модуля для выбранного объекта, в котором необходимо выбрать событие, для которого требуется создать процедуру обработки, в списке, расположенном в верхнем правом углу

окна модуля, и ввести текст процедуры.

Таблица 1.1 – Элементы управления

Элемент управления	Имя	Назначение	Кнопка для создания
Надпись	Label	Позволяет отобразить в форме неизменяемый текст, например подпись к рисунку	
Поле	TextBox	Для ввода или вывода данных	
Кнопка	CommandButton	Создает кнопку, при нажатии которой выполняется команда	
Флажок	CheckBox	Создает ячейку, которая может быть помечена пользователем и используемая для предоставления выбора нескольких вариантов	
Переключатель	OptionButton	Используется для предоставления выбора одного варианта из многих	
Выключатель	ToggleButton	Создает кнопку, имеющую два состояния: включено и выключено	
Полоса прокрутки	ScroolBar	Создает графический инструмент для быстрого перемещения по длинным спискам элементов, отображающий текущее положение	
Счётчик	Spin Button	Прокручивающий элемент управления используется совместно с другими элементами для увеличения или уменьшения значений	
Рисунок	Image	Отображает в форме точечный рисунок, значок или метафайл	
Список	ListBox	Вставляет список выбираемых пользователем элементов	
Поле со списком	ComboBox	Содержит вводимый и изменяемый пользователем текст	
Рамка	Frame	Позволяет установить графическую или функциональную группировку элементов управления	
Вкладки	MultiPage	Служит для представления нескольких экранов информации в виде единого набора	
Ярлыки	TabStrip	Позволяет создать несколько страниц в одной и той же области окна	

Свойства объектов. Каждый объект обладает некоторыми характеристиками, или свойствами. Изменяя свойства, можно менять характеристики объекта. Таким образом, *свойство* представляет собой атрибут объекта, определяющий его характеристики, такие как размер, цвет, положение на экране и состоя-

ние объекта, например доступность или видимость.

Синтаксис применения свойства **Объект.Свойство.**

Основные общие свойства элементов управления приведены в таблице 1.2.

Таблица 1.2 – Общие свойства элементов управления

Свойство	Описание
Caption	Надпись, отображаемая при элементе управления
AutoSize	Допустимые значения: True (устанавливает режим автоматического изменения размеров элемента управления так, чтобы на нем полностью помещался текст, присвоенный свойству Caption) и False (в противном случае)
Visible	Допустимые значения: True (элемент управления отображается во время выполнения программы) и False (в противном случае)
Enabled	Допустимые значения: True (пользователь вручную может управлять элементом управления) и False (в противном случае)
Height и Width	Устанавливают геометрические размеры объекта (высоту и ширину)
Left и Top	Устанавливают координаты верхнего левого угла элемента управления, определяющие его местоположение в форме
ControlTipText	Устанавливает текст в окне всплывающей подсказки, связанной с элементом управления. Пример: <code>CommandButton1.ControlTipText = "Это кнопка"</code>
BackColor, ForeColor, BorderColor	Устанавливают цвет заднего и переднего плана элемента управления, также его границы
BackStyle	Устанавливает тип заднего фона
BorderStyle	Устанавливает тип границы. Допустимые значения: <code>fmBorderStyleSingle</code> (граница в виде контура); <code>fmBorderStyleNone</code> (граница невидима)
Picture (создание картинки)	Внедряет картинку на элемент управления. Например, на поверхности кнопки картинка отображается с помощью следующей инструкции: <code>CommandButton1.Picture = LoadPicture("c:\my doc\Круг.bmp")</code>
Picture (удаление картинки)	После того как картинка создана на элементе управления, иногда возникает необходимость ее удалить. Это легко достигается присвоением свойству Picture значения <code>LoadPicture("")</code>

Метод. Объект содержит также список методов, которые к нему могут быть применены. Например, показать форму на экране или убрать его можно с помощью методов Show и Hide соответственно. Таким образом, метод представляет собой действие, выполняемое над объектом.

Синтаксис применения метода **Объект.Метод.**

В таблице 1.3 приведены общие методы элементов управления.

Событие представляет собой действие, распознаваемое объектом (например, щелчок мышью или нажатие клавиши), для которого можно запрограммировать отклик. События возникают в результате действий пользователя программы или же они могут быть вызваны системой (например, процедуры Initialize, Load, Click и DblClick). Эти процедуры имеют следующий *синтаксис*:

Sub UserForm_Событие()
 Последовательность инструкций
End Sub

Таблица 1.3 – Основные общие методы элементов управления

Метод	Описание
Add	Позволяет добавить элемент управления во время выполнения программы
Move	Перемещает элемент управления
SetFocus	Устанавливает фокус на вызвавшем этот метод элементе управления
Zorder	Помещает объект до или после всех пересекающихся с ним объектов

В таблице 1.4 приведены события элементов управления, для которых можно создать процедуры обработки событий. Каждый элемент управления, который вы добавите в свою форму, будет иметь доступ к этим событиям.

Таблица 1.4 – События элементов управления

Событие	Описание
Click	Происходит, когда пользователь выбирает элемент управления с помощью одинарного щелчка кнопкой мыши
DblClick	Происходит, когда пользователь выбирает элемент управления с помощью двойного щелчка кнопкой мыши
Change	Происходит при изменении значения элемента управления
GotFocus	Происходит, когда элемент управления получает фокус
LostFocus	Происходит, когда элемент управления теряет фокус
Error	Используется при уведомлении об ошибке
Exit	Происходит, когда с элемента управления снимается выделение

Отладка программ – это проверка и внесение исправлений в программу при ее разработке. В процессе отладки программы возможны три вида ошибок:

1) *ошибки компиляции*, возникающие при неправильном использовании синтаксиса инструкций, свойств и методов объектов. Например, при некорректном вводе числа скобок, неправильном имени, неполном вводе инструкции и т. д. Некоторые из этих ошибок обнаруживаются VBA при завершении набора строки с инструкцией в редакторе кода и после нажатия клавиши **Enter**. Строка, в которой содержится ошибка, выделяется красным цветом, и на экране отображается диалоговое окно с сообщением о возможной причине, вызвавшей ошибку (рисунок 1.2). Такие ошибки выявляются на уровне компиляции и легко исправляются;

2) *ошибки выполнения*, возникающие после успешной компиляции программы при ее выполнении. Причиной таких ошибок может быть отсутствие данных, неправильные данные, введенные пользователем (строка вместо числа, точка вместо запятой или наоборот и т. д.), некорректность вычислений (деление на ноль), некорректная информация при считывании диска и т. д. В этих и

подобных случаях на экране отображается диалоговое окно с сообщением о номере ошибки и возможной причине, ее вызвавшей (рисунок 1.3).

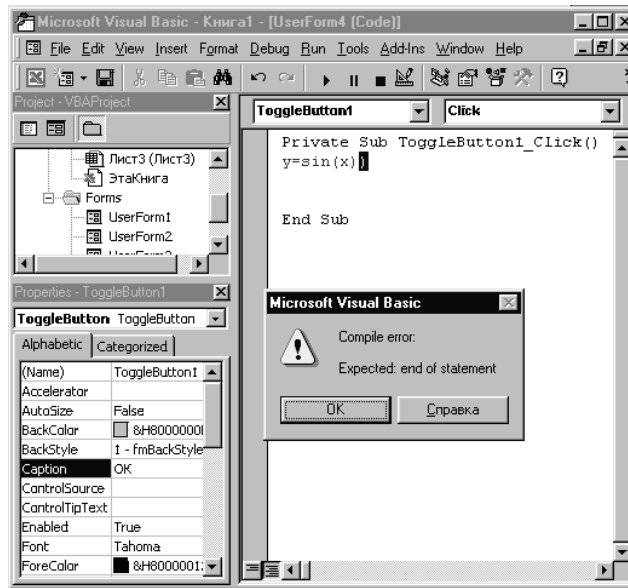


Рисунок 1.2 – Ошибка компиляции

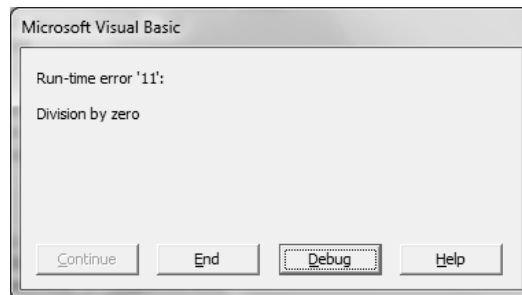


Рисунок 1.3 – Сообщение об ошибке выполнения в диалоговом окне

Если в ДО **Microsoft Visual Basic** нажать кнопку **Отладка (Debug)**, то в строке модуля желтым цветом будет выделена строка, вызвавшая ошибку и по причине которой выполнение программы было прервано. Кроме того, эта строка будет помечена стрелочкой. При прерывании выполнения программы VBA переходит в *режим прерывания*. Одной из наиболее удобных возможностей режима прерывания являются возможность узнать текущее значение переменных и свойств. Для этого достаточно расположить указатель мыши на имени свойства или переменной. Это вызовет появление всплывающей подсказки с текущим значением переменной или свойства. Для устранения возникновения этих ошибок целесообразно использовать имеющиеся в VBA средства обработки ошибок;

3) *логические ошибки*, являющиеся причиной неправильной работы программы. Для нахождения таких ошибок предназначены средства отладки, которые позволяют обнаружить логические ошибки и ошибки периода выполнения, а также наблюдать за выполнением программы.

При составлении приложений важно предусмотреть, чтобы программа

анализировала возможные ошибки, возникающие при ее выполнении по вине пользователя, и информировала его об этом, подсказывая, что конкретно он сделал неправильно.

Обработка ошибок – это задание реакции на ошибки, которые возникают во время выполнения программы. Целесообразно в программе создать подпрограмму – обработчик ошибок, которые могут возникнуть в данной программе на этапе выполнения.

К средствам обработки ошибок относятся операторы **On Error** и **Resume**.

Оператор **On Error** осуществляет передачу управления на подпрограмму обработки ошибок. Возможные варианты синтаксиса оператора представлены в таблице 1.5.

Оператор **Resume** передает управление из обработчика ошибок в программу, возможные варианты синтаксиса оператора представлены в таблице 1.5.

Таблица 1.5 – Варианты синтаксиса операторов обработки ошибок

Синтаксис	Действие
On Error GoTo метка	Передача управления на подпрограмму, идентифицирующуюся меткой
On Error Resume Next	Ошибка игнорируется, и управление передается следующему оператору за тем, при выполнении которого возникла ошибка
On Error GoTo 0 -	Отключает обработку ошибок для данной процедуры
Resume [0] -	Повторное выполнение оператора, вызвавшего ошибку
Resume Next -	Выполнение следующего оператора за тем, при выполнении которого возникла ошибка
Resume метка -	Выполнение оператора, помеченного меткой

Функции проверки типов (таблица 1.6) проверяют, является ли переменная выражением специфицированного типа. Возвращают значение True, если переменная имеет заданный тип, и False – в противном случае.

Таблица 1.6 – Функции проверки типов

Функция	Проверяет...
IsArray(переменная)	... является ли переменная массивом
IsDate(переменная)	... является ли переменная датой
IsEmpty(переменная)	... была ли переменная описана инструкцией Dim
IsError(переменная)	... является ли переменная кодом ошибки
IsNull(переменная)	... является ли переменная пустым значением (Null)
IsNumeric(переменная)	... является ли переменная числовым значением
IsObject(переменная)	... является ли переменная объектом

Рассмотрим процесс создания приложения (рисунок 1.4), в котором предотвращается появление ошибки на конкретном примере, в котором программа производит деление числителя на знаменатель по нажатию кнопки **Счет**

без контроля появления возможных ошибок.

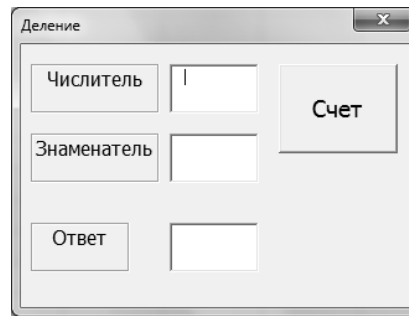


Рисунок 1.4 – Пользовательская форма

Текст программы приведен ниже.

```
Private Sub CommandButton1_Click()
Dim Числитель As Single, Знаменатель As Single, Результат As Single
    Числитель = CDb1(TextBox1.Text)
    Знаменатель = CDb1(TextBox2.Text)
    Результат = Числитель / Знаменатель
    TextBox3.Text = CStr(Результат)
End Sub
```

Если пользователь по невнимательности забудет ввести в поле *Числитель* или в поле *Знаменатель* число, то при нажатии кнопки **Счет** происходит аварийное прерывание программы с сообщением о несоответствии типов, отображаемом в диалоговом окне (рисунок 1.5).

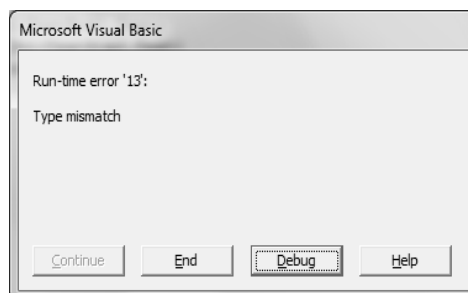


Рисунок 1.5 – ДО с сообщением об ошибке

Данное сообщение об ошибке связано с одной из следующих инструкций в программе: `Числитель = CDb1(TextBox1.Text)` или `Знаменатель = CDb1(TextBox2.Text)`, где аргументом функции должна быть строка, преобразуемая в число. Если в какое-то из полей, *Числитель* или *Знаменатель*, ничего не введено, по умолчанию из этого поля будет считываться пустая строка. Но пустая строка не может быть преобразована в число, и поэтому из-за функции преобразования типов `CDbl` происходит ошибка. Ошибка о несоответствии типов возникнет также, если в одно из полей пользователь по неосторожности введет число с десятичной запятой, а установками системы предусматривается

десятичная точка и наоборот.

Такие ошибки ввода легко избежать, если производить в программе предварительную проверку того, будет ли вводимая информация в поля ввода преобразовываться в числа и если вводимая информация не преобразуется в числа, то выдается сообщение о поле, в которое некорректно введены данные, и на него перемещается фокус. Процедура предварительной проверки записана ниже.

```
Private Sub CommandButton1_Click()
Dim Числитель As Single, Знаменатель As Single, Результат As Single
If IsNumeric(TextBox1.Text) = False Then
MsgBox "Ошибка в числителе", 32
TextBox1.SetFocus
Exit Sub
End If
If IsNumeric(TextBox2.Text) = False Then
MsgBox "Ошибка в знаменателе", 32
TextBox2.SetFocus
Exit Sub
End If
Числитель = CDb1(TextBox1.Text)
Знаменатель = CDb1(TextBox2.Text)
Результат = Числитель / Знаменатель
TextBox3.Text = CStr(Результат)
End Sub
```

Но и это еще не все. Если пользователь в поле *Знаменатель* введет 0, то также произойдет аварийная остановка выполнения программы с отображением в диалоговом окне сообщения: **Division by zero (Деление на 0)**. Для избежания подобной ошибки будем проверять не только, являются ли введенные в поле данные числом, но и что это не ноль. Например, добавим перед расчетным блоком в процедуре следующую дополнительную проверку:

```
If CSng(TextBox2.Text) = 0 Then
MsgBox "Знаменатель не может быть нулем", 32
TextBox2.SetFocus
Exit Sub
End If
```

Но если пользователь введет в поле *Знаменатель*, например, значение 40, то произойдет аварийная остановка выполнения программы с отображением ошибки переполнения (рисунок 1.6).

В идеале разрабатываемое приложение не должно никогда аварийно прерываться. В приложении следует создать средства перехвата возможной ошибки, обработать её, выдать сообщение пользователю и обеспечить безаварийное продолжение работы приложения. Это можно реализовать с помощью инструкции `On Error`, которая производит перехват ошибки и устанавливает, что программа должна делать в случае появления ошибки:

– пользователь информируется программой в случае появления ошибки переполнения, знаменателю и числителю присваиваются 1 и с этими данными проводятся вычисления.

– при появлении ошибки, отличной от ошибки переполнения, выполнение программы прерывается с информированием пользователя об ошибке (рисунок 1.7).

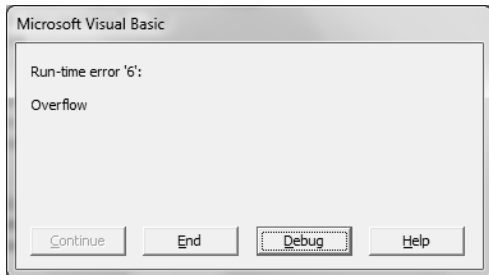


Рисунок 1.6 – ДО с сообщением об ошибке

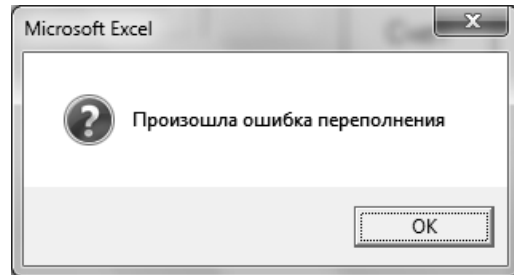


Рисунок 1.7 – ДО с указанием типа ошибки

Полный текст программы:

```
Private Sub CommandButton1_Click()
Dim Числитель As Single, Знаменатель As Single, Результат As Single
On Error GoTo Обработка
If IsNumeric(TextBox1.Text) = False Then
MsgBox "Ошибка в числителе", 32
TextBox1.SetFocus
Exit Sub
End If
If IsNumeric(TextBox2.Text) = False Then
MsgBox "Ошибка в знаменателе", 32
TextBox2.SetFocus
Exit Sub
End If
If CSng(TextBox2.Text) = 0 Then
MsgBox "Знаменатель не может быть нулем", 32
TextBox2.SetFocus
Exit Sub
End If
Числитель = CDb1(TextBox1.Text)
Знаменатель = CDb1(TextBox2.Text)
Результат = Числитель / Знаменатель
TextBox3.Text = CStr(Результат)
Exit Sub
Обработка: Select Case Err.Number
Case Is = 6
MsgBox " Произошла ошибка переполнения", 32
TextBox1.Text = 1
TextBox2.Text = 1
Знаменатель = 1
Числитель = 1
```

```

Resume
Case Else
MsgBox "Произошла ошибка: ", 32
Exit Sub
End Select
End Sub

```

В случае, если разрабатываемое приложение состоит из нескольких процедур, причем в некоторых из них необходимо создать по обработчику ошибок, бывает более удобно для сокращения программы и для большей ясности структуры кода написать отдельную процедуру с обработчиком всех ошибок.

1.2 Задание к лабораторной работе

Создать пользовательскую форму (далее – ПФ) для вычисления по вариантам из таблицы 1.7, предусмотрев на ней функцию очистки окон, кнопку выхода из программы и функцию обработки ошибок.

Таблица 1.7 – Варианты заданий

Номер варианта	Задание
1	Вычислить $f(x, y) = x - y - (1 + 2x)^y + 3^{\sqrt{ y-x }}$
2	Вычислить $f(x, y, z) = \frac{x + y + z}{x^2 + y^2 + z^2}$
3	Вычислить площадь прямоугольного треугольника по двум катетам
4	Вычислить $f(x, y) = \frac{x}{1+y} + \frac{y}{1+x} + \frac{y}{y+x}$
5	Вычислить площадь куба по его стороне
6	Вычислить длину гипотенузы по заданным длинам катетов
7	Вычислить дискриминант квадратного уравнения по заданным коэффициентам
8	Вычислить произведение четырех вещественных чисел
9	Вычислить $f(x, y, z) = \frac{x + y + z}{x \cdot y \cdot z}$
10	Вычислить $f(x, y) = (x + y)(x^2 + y^2)(x^3 + y^3)$
11	Вычислить объем шара по заданному радиусу
12	Вычислить объем цилиндра по заданному радиусу и высоте
13	Вычислить $f(x, y, z) = \frac{x \cdot y \cdot z}{x + y^2 + z^3}$
14	Вычислить $f(x, y, z) = \frac{\sqrt{x + 2y + 3z}}{x^3 + y^2 + z}$

1.3 Пример выполнения задания

Создать пользовательскую форму для вычисления суммы $a + b = c$.

Порядок выполнения работы.

- 1 Запустить редактор VBA и выполнить команду *Insert* → *UserForm*.
- 2 Поместить на ПФ элементы управления, как показано на рисунке 1.8.

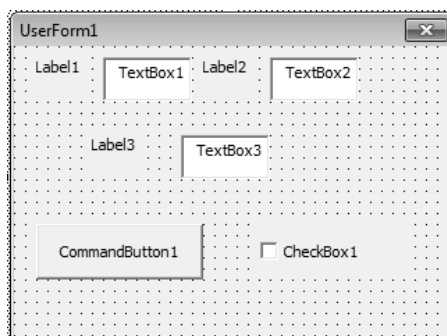


Рисунок 1.8 – Вид пользовательской формы

3 Изменить свойства объектов (таблица 1.8) на ПФ с помощью окна свойств. Вид ПФ после назначения свойств представлен на рисунке 1.9.

Таблица 1.8 – Свойства объектов

Свойство	Значение
Label1.Caption	<i>a</i>
Label2.Caption	<i>b</i>
Label3.Caption	<i>c</i>
CommandButton1.Caption	Результат
CheckBox1.Caption	Очистка окон
Для всех объектов *.BackColor	По своему вкусу выбрать цвет из палитры
Для Label1, Label2, Label3 свойство *.Font	В ДО «Шрифт» выбрать размер 16

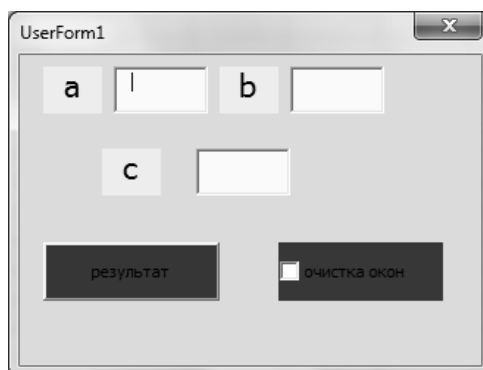


Рисунок 1.9 – Вид пользовательской формы после назначения свойств

4 Написать программный код. Для этого рекомендуется выполнить двойной щелчок по кнопке *результат* и перейти в окно программы, где набрать текст процедуры обработки события Click() для кнопки CommandButton1:

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

5 Двойной щелчок по элементу управления CheckBox1 вызовет процедуру обработки события Click(), где необходимо написать программный код для очистки полей TextBox:

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

Контрольные вопросы

- 1 Перечислите элементы управления в VBA.
- 2 Перечислите общие свойства элементов управления.
- 3 Перечислите свойства элемента управления TextBox.
- 4 Перечислите свойства элемента управления Label.
- 5 Назначение элемента управления TextBox.
- 6 Назначение элемента управления Label.
- 7 Назначение элемента управления CommandButton.
- 8 Перечислите свойства управления CommandButton.
- 9 Перечислите свойства пользовательской формы.
- 10 Каков синтаксис применения свойств?

2 Лабораторная работа № 13. Программирование на алгоритмическом языке

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

2.1 Теоретические сведения

Элемент управления *OptionButton* (переключатель) позволяет выбрать один из нескольких взаимоисключающих параметров или действий. Переключатели обычно отображаются группами, созданными с помощью элемента управления *Frame*, обеспечивая возможность выбора варианта.

Элемент управления *Frame* (рамка) используют для группировки объектов. Например, на форме может быть расположено несколько групп элементов управления, выполняющих различные функции. Если на форме уже есть элементы управления, то они не попадут в группу, если поместить поверх них *Frame*. Для их группировки необходимо сначала создать на форме рамку и перетащить на нее остальные элементы управления. Как правило, рамкам не назначают обработчики событий, используя их как контейнеры для других объектов. Для обращения к элементу управления, который расположен на рамке, используют такой же синтаксис, как и для доступа к объекту, расположенному на форме.

Основными событиями переключателя являются события *Click* и *Change*. Наиболее часто используемые свойства элемента управления *OptionButton* представлены в таблице 2.1.

Таблица 2.1 – Свойства элемента управления *OptionButton*

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

2.2 Задания к лабораторной работе

Создать пользовательскую форму согласно варианту, используя переключатели. Предусмотреть функцию очистки и кнопку выхода из приложения.

Вариант 1

1 Выбирается число от 1 до 4, определяющее пору года. Вывести название этой поры.

2 На трех заводах «Альфа», «Плутон» и «Рубин» иногда происходят аварии, сведения о которых за последние пять лет фиксируются на листе Excel.

Для выбранного пользователем завода и года вывести число аварий.

Вариант 2

1 Выбирается число от 1 до 7, определяющее день недели. Дать название этого дня.

2 Известен состав из пяти учредителей (акционеров) АО «Рога и копыта». Оформить лист, где будут указаны сведения о составе учредителей: ФИО учредителя, количество обыкновенных акций, количество привилегированных акций. Для выбранного учредителя и вида акции вывести общую стоимость акций данного вида. Стоимости одной акции каждого вида известны и должны задаваться в отдельных полях ввода.

Вариант 3

1 Выбирается число от 1 до 5. Дать название этого числа.

2 Подготовить лист, где будут приведены цены для пяти наименований товаров. Для выбранного товара вывести цену и цену со скидкой. Величина скидки (в процентах) задается.

Вариант 4

1 Вводится нецелое число. Вывести либо его целую часть, либо дробную в зависимости от выбора пользователя.

2 Разместить на листе сведения о заработной плате работников некоторого предприятия (пять работников): ФИО работника, оклад. Для выбранного работника рассчитать подоходный налог (9 % от оклада за вычетом минимального размера оплаты труда), пенсионный налог (1 % от оклада), профсоюзный взнос (1 % от оклада), сумму к выдаче. Минимальный размер оплаты труда вводить в поле ввода.

Вариант 5

1 Банк предлагает три вида срочных вкладов: на 3 месяца под 27 %, на 6 месяцев под 29 % и на год под 30 %. Вкладчик положил N у. е. на один из срочных вкладов. Создать приложение для расчета суммы выплаты по вкладу.

2 В таблице приведены сведения о количестве легковых автомобилей, выпущенных ведущими мировыми производителями в первом полугодии текущего года по месяцам. Для выбранного пользователем производителя и месяца вывести число автомобилей, произведенных за этот месяц.

Вариант 6

1 Задано расстояние в метрах. Пересчитать это расстояние в километрах, милях, футах или ярдах на выбор пользователя (1 миля = 1,609 км, 1 м = 1,094 ярда, 1 м = 3,281 фута).

2 Подготовить приложение для расчета стоимости выбранной марки автомобиля в зависимости от его дополнительной комплектации.

Вариант 7

1 Дан объем в литрах. Пересчитать этот объем в пинтах, галлонах, бушелях и квартах (английские меры объема жидких и сыпучих тел) на выбор пользователя (1 л = 1,706 пинты, 1 л = 0,220 галлона, 1 бушель = 36,35 л, 1 кварта = 1,136 л).

2 Создать приложение для расчета стоимости железнодорожного билета в зависимости от направления, типа вагона и сезона (летом стоимость увеличивается на 20 %, зимой уменьшается на 10 %).

Вариант 8

1 Дана масса в килограммах. Пересчитать эту массу в пудах, фунтах, центнерах или тоннах на выбор пользователя (1 пуд = 16,38 кг, 1 фунт = 0,409 кг, 1 т = 1000 кг, 1 ц = 100 кг).

2 Подготовить расчет стоимости санаторного лечения в зависимости от срока пребывания (7, 12, 15 и 24 дня) и полноты питания (завтрак; завтрак и обед; завтрак, обед и ужин).

Вариант 9

1 Дано расстояние в метрах. Пересчитать его в верстах, саженьях, аршинах или вершках на выбор пользователя (1 верста = 1,067 км, 1 сажень = 2,134 м, 1 аршин = 0,7112 м, 1 вершок = 4,445 см).

2 Вывести название книг с ценой. Указать варианты доставки: курьерская (постоянная цена N р.), наложенным платежом (зависит от количества книг и от общей стоимости) и оплата через банк (зависит от стоимости книг). Написать приложение для выбора книг и подсчета общей стоимости.

Вариант 10

1 Разработать пользовательскую форму для нахождения периметра или площади прямоугольника по заданным сторонам.

2 Создать приложение для расчета стоимости авиабилета в зависимости от направления, расположения кресла (в середине, у прохода, у окна) и типа салона (для курящих или не курящих).

Вариант 11

1 Разработать пользовательскую форму для нахождения периметра или площади треугольника по заданным сторонам.

2 Вывести список изделий из мебели (комод, шифоньер, тумбочка, ...) с ценой изготовления. Указать вид материала. Создать приложение для расчета стоимости покупки выбранного набора мебели.

Вариант 12

1 Банк предлагает три вида долгосрочных вкладов: на 1 год под 5 %, на 2 года под 7 % и на 5 лет под 10 %. Вкладчик положил N у. е. на один из срочных вкладов. Создать приложение для расчета суммы выплаты по вкладу.

2 Вывести список товаров с ценой. Указать варианты доставки: курьерская (постоянная цена N р.), наложенным платежом (зависит от количества товаров и от общей стоимости) и оплата через банк (зависит от стоимости товаров). Написать приложение для выбора товаров и подсчета общей стоимости.

Вариант 13

1 Разработать пользовательскую форму для нахождения длины окружности или площади круга по введенному радиусу.

2 Вывести список тренажеров, которые имеются в тренажерном зале с ценой. Указать время посещения зала (утром, днем, вечером, выходные дни). Написать приложение для посещения зала с подсчетом общей стоимости, кото-

рая зависит от времени, для утра и дня – скидки, самое дорогое время – выходные, самое дешевое – утро.

Вариант 14

1 Выбирается число от 1 до 7, определяющее день недели. Дать название этого дня (рабочий день или выходной).

2 Создать пользовательскую форму для расчета суммы банковских валютных операций (купля или продажа, вид валюты).

2.3 Пример выполнения задания

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

Порядок выполнения работы.

1 Запустить редактора VBA и выполнить команду *Insert* → *UserForm*.

2 Поместить на форму элементы, требуемые для решения задачи, с панели элементов и расположить их нужным образом (рисунок 2.1).

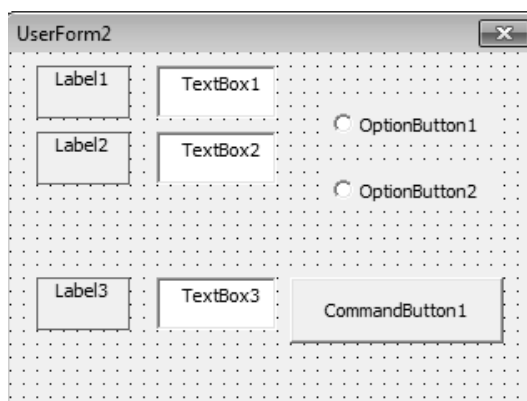


Рисунок 2.1 – Вид пользовательской формы

3 Изменить свойства объектов (таблица 2.2) на форме с помощью окна свойств. Вид формы после назначения свойств представлен на рисунке 2.2.

Таблица 2.2 – Свойства объектов

Свойство	Значение свойства
Label1.Caption	а
Label2.Caption	в
Label3.Caption	с
CommandButton1.Caption	Расчет
OptioButton1.Caption	Сложение
OptioButton2.Caption	Вычитание
Для всех объектов свойство *.BackColor	По своему вкусу выбрать цвет из палитры цветов
Для Label1, Label2, Label3 свойство .Font	В ДО “Шрифт” выбрать размер 16

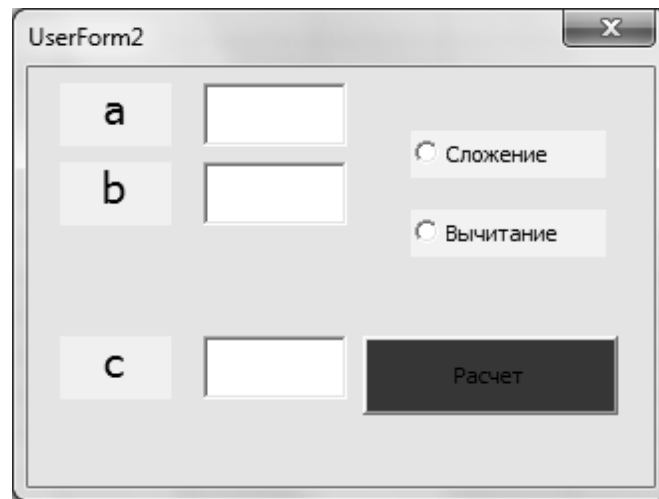


Рисунок 2.2 – Вид пользовательской формы после назначения свойств

4 Написать программный код. Для этого рекомендуется выполнить двойной щелчок по кнопке *Расчет* и перейти в окно программы, где набрать текст процедуры обработки события Click() для кнопки CommandButton1:

```
Private Sub CommandButton1_Click()
Dim a As Integer, b As Integer, c As Integer
a = TextBox1.Value
b = TextBox2.Value
If OptionButton1.Value = True Then
    c = a + b
End If
If OptionButton2.Value = True Then
    c = a - b
End If
TextBox3.Value = c
End Sub
```

Контрольные вопросы

- 1 Перечислите свойства элемента управления OptionButton.
- 2 Назначение элемента управления OptionButton.
- 3 Назначение элемента управления Frame.

3 Лабораторная работа № 14. Программирование на алгоритмическом языке

Цель работы: изучить свойства, события и методы элемента управления *ListBox* (список); научиться использовать списки при решении задач.

3.1 Теоретические сведения

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

Наиболее часто используемые свойства элемента управления *ListBox* представлены в таблице 3.1, а методы – в таблице 3.2.

Таблица 3.1 – Свойства элемента управления *ListBox*

Свойство	Описание
ListIndex	Возвращает номер текущего элемента списка. Нумерация элементов списка начинается с нуля
ListCount	Возвращает число элементов списка
TopIndex	Возвращает элемент списка с наибольшим номером
ColumnCount	Устанавливает число столбцов в списке
TextColumn	Устанавливает столбец в списке, элемент которого возвращается свойством Text
Enabled	Допустимые значения: True (запрещен выбор значения из списка пользователем) и False (в противном случае)
Text	Возвращает выбранный в списке элемент
List	Возвращает элемент списка, стоящий на пересечении указанных строки и столбца. <i>Синтаксис: List (row, column)</i>
RowSource	Устанавливает диапазон, содержащий элементы списка
ControlSource	Устанавливает диапазон (ячейку), куда возвращается выбранный элемент из списка
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 последовательных элементов списка)
Selected	Допустимые значения: True (если элемент списка выбран) и False (в противном случае). Используется для определения выделенного текста, когда свойство MultiSelect имеет значение fmMultiSelectMulti или fmMultiSelectExtended

Заполняется объект *ListBox* только программно!

При этом используются следующие способы заполнения:

– поэлементно;

- массивом;
- из диапазона на листе Excel, в который предварительно введены элементы списка.

Таблица 3.2 – Методы элемента управления *ListBox*

Метод	Описание
Clear	Удаляет все элементы из списка
RemoveItem	Удаляет из списка элементы с указанным номером. <i>Синтаксис: Remove Item (index)</i> index – номер удаляемого из списка элемента
AddItem	Добавляет элемент в список. <i>Синтаксис: AddItem ([Item ,[VarIndex]])</i> item – элемент, добавляемый в список; varindex – номер добавляемого элемента

Примеры заполнения списков.

1 Поэлементно, если список состоит из одной колонки:

```
With ListBox1
    .AddItem "Июнь"
    .AddItem "Июль"
    .AddItem "Август"
End With
```

2 Массивом, если список состоит из одной колонки:

```
With ListBox1
    .List = Array("Июнь", "Июль", "Август")
    .ListIndex = 1
End With
```

3 Из диапазона A1 : B4, в который предварительно введены элементы списка. Результат выбора (индекс выбранной строки) выводится в ячейку C1:

```
With ListBox1
    .ColumnCount = 2
    .RowSource = "A1:B4"
    .ControlSource = "C1"
End With
```

4 Поэлементно, если список состоит из нескольких колонок, например двух:

```
With ListBox1
    .ColumnCount = 2
    .AddItem "Июнь"
    .List(0, 1) = "Сессия"
    .AddItem "Июль"
```

```

.List(1, 1) = "Каникулы"
.AddItem "Август"
.List (2, 1) = "Каникулы"
End With

```

5 Массивом, если список состоит из нескольких колонок, например двух:

```

Dim A (2, 1) As String
A(0, 0) = "Июнь"
A(0, 1) = "Сессия"
A(1, 0) = "Июль"
A(1, 1) = "Каникулы"
A(2, 0) = "Август"
A(2, 1) = "Каникулы"
With ListBox1
.ColumnCount = 2
.List = A
End With

```

3.2 Задания к лабораторной работе

Разработать пользовательскую форму, содержащую список.

Вариант 1

Дан одномерный массив. Отсортировать его по возрастанию элементов. Вывести в один список – исходный массив, в другой – отсортированный.

Вариант 2

Дан одномерный массив. Заменить четные числа на 1, нечетные – на –1. Вывести в один список – исходный массив, в другой – преобразованный.

Вариант 3

Дан одномерный массив. Вывести в один список – исходный массив, в другой – только элементы, кратные трем.

Вариант 4

Вычислить $\frac{x^2}{2}, \frac{x^3}{3}, \dots, \frac{x^{11}}{11}$ для указанного значения x .

Вариант 5

Вывести члены арифметической прогрессии. Значение первого члена, разность и количество членов задаются.

Вариант 6

Вывести на пользовательскую форму члены геометрической прогрессии. Значение первого члена, знаменатель и количество членов задаются (формула n -го члена $b_n = b_1 q^{n-1}$).

Вариант 7

Разработать программу, содержащую многостолбцовый список. Рассчитать таблицу значений функции $y = \sqrt{x^2 + k^2}$, где x меняется от –2 до 2 с шагом 0,1, а k – параметр, задаваемый пользователем. Таблицу поместить в двухстолбцовый список.

Вариант 8

Составить таблицу перевода километров в мили на интервале от 10 до 50 с шагом 10 (1 миля = 1,609 км). Таблицу поместить в двухстолбцовый список.

Вариант 9

Составить таблицу перевода метров в ярды на интервале от 2 до 10 с шагом 0,5 (1 м = 1,094 ярда). Таблицу поместить в двухстолбцовый список.

Вариант 10

Составить таблицу перевода метров в футы на интервале от 10 до 50 с шагом 5 (1 м = 3,281 фута). Таблицу поместить в двухстолбцовый список.

Вариант 11

Составить таблицу квадратных корней из чисел от a до b с шагом 0,1. Значения a и b задаются ($a < b$). Таблицу поместить в двухстолбцовый список.

Вариант 12

Дан массив размерностью $n \times m$. Найти максимальный по модулю элемент массива. Вывести массив в список и найденный элемент в поле.

Вариант 13

Дан массив размерностью $n \times m$. Найти минимальный элемент массива. Вывести массив в список и найденный элемент в поле.

Вариант 14

Дан массив размерностью $n \times m$. Найти индекс минимального элемента массива. Вывести массив в список и найденный индекс в поле.

3.3 Пример выполнения задания

Создать приложение, которое позволит подсчитать сумму или произведение выбранных в списке чисел.

1 Перейдем в VBA и, выполнив команду Insert → UserForm. Расположим на форме элементы управления, как показано на рисунке 3.1.

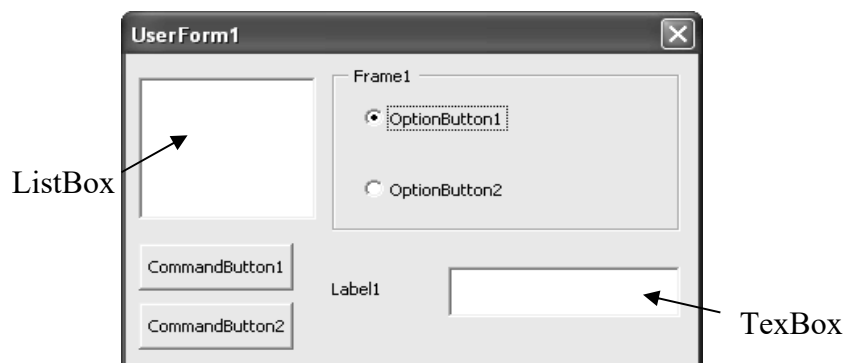


Рисунок 3.1 – Проектируемая пользовательская форма

Назначение размещенных на форме элементов управления.

CommandButton1 – нажатие на кнопку запускает процедуру обработки события (Private Sub CommandButton1_Click()), которое определяет, какой переключатель выбран. В зависимости от выбранного переключателя производится

действие над выбранными в списке числами. Найденное значение выводится в поле TextBox1.

CommandButton2 – нажатие на кнопку запускает процедуру обработки события (Private Sub CommandButton2_Click()), которая закрывает диалоговое окно.

ListBox1(список) – список для ввода чисел.

TextBox1 (поле) – в это поле будет выводиться результат. Поле сделаем недоступным для пользователя, т. е. пользователь не сможет ни ввести, ни скорректировать данные в этом поле.

Label1 (надпись) – пояснительная надпись для поля вывода.

Frame1 (рамка) – используется для группировки переключателей.

OptionButton1 и OptionButton2 – выбор переключателя указывает, какая операция будет выполняться над выбранными числами.

В модуле формы набираем код программы:

```
Private Sub CommandButton1_Click()
Dim i As Integer
Dim n As Integer
Dim Сумма As Double
Dim Произведение As Double
Dim Результат As Double
If OptionButton1.Value = True Then
Сумма = 0
With ListBox1
For i = 0 To .ListCount - 1
If .Selected(i) = True Then
Сумма = Сумма + .List(i)
End If
Next i
End With
Результат = Сумма
End If
If OptionButton2.Value = True Then
Произведение = 1
With UserForm1.ListBox1
For i = 0 To .ListCount - 1
If .Selected(i) = True Then
Произведение = Произведение * .List(i)
End If
Next i
End With
Результат = Произведение
End If
TextBox1.Text = CStr(Результат)
End Sub
Private Sub CommandButton2_Click()
UserForm1.Hide
End Sub
Private Sub UserForm_Initialize()
With ListBox1
```

При выборе первого переключателя вычисляется сумма выбранных элементов

При выборе второго переключателя вычисляется произведение выбранных элементов

Результат выводится в поле TextBox1

Процедура закрытия диалогового окна

Процедура инициализации диалогового окна

```

.List = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
.ListIndex = 0
.MultiSelect = fmMultiSelectMulti
End With
With UserForm1.OptionButton1.Value = True
.Caption = "Сумма"
.ControlTipText = "Сумма выбранных элементов"
End With
OptionButton2.ControlTipText = "Произведение
выбранных элементов"
CommandButton2.ControlTipText = "Выход из про-
граммы"
CommandButton1.ControlTipText = "Нахождение
результата"
UserForm1.Caption = "Операции над элементами
списка"
OptionButton2.Caption = "Произведение"
Label1.Caption = "Результат"
CommandButton1.Caption = "Вычислить"
CommandButton2.Caption = "Отмена"
Frame1.Caption = "Операция"
TextBox1.Enabled = False
End Sub

```

Заполнение списка

Установка режима выбо-
ра (при загрузке формы
первоначально будет вы-
бран переключатель
«Сумма»)

Задание текста всплыва-
ющих подсказок у эле-
ментов управления

Задание заголовка поль-
зовательской формы

Задание видимых надпи-
сей для объектов

Инструкция делает Text-
Box1 недоступным для
пользователя

После конструирования формы и написания кода в модуле формы выберем команду Run. На экране появится форма, представленная на рисунке 3.2

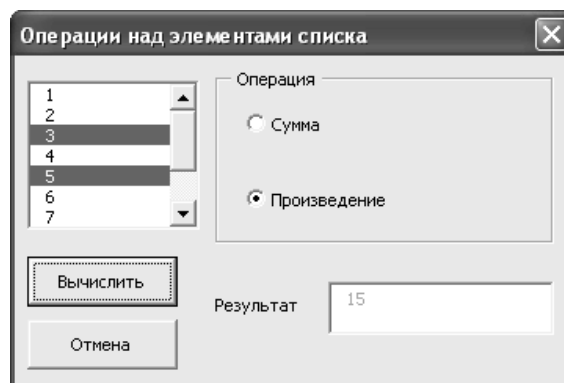


Рисунок 3.2 – Спроектированная форма

Контрольные вопросы

- 1 Перечислите свойства элемента управления ListBox.
- 2 Назначение элемента управления ListBox.
- 3 Способы заполнения ListBox.
- 4 Как обратиться к элементам ListBox?
- 5 Перечислите методы ListBox.

4 Лабораторная работа № 15. Программирование на алгоритмическом языке

Цель работы: изучить основные свойства и методы элемента управления *Image* (рисунок); построить график на листе Excel и на пользовательской форме.

4.1 Теоретические сведения

В Excel различаются два вида диаграмм: внедренная на рабочий лист (объект **ChartObject**) и созданная на специальном листе для диаграмм (объект **Charts**). Свойства и методы этих объектов представлены в таблицах 4.1 и 4.2 соответственно.

Таблица 4.1 – Свойства объектов ChartObject и Charts

Свойство	Значение
ChartArea	Возвращает объект ChartArea – область на листе, отведенная под диаграмму. В следующем примере устанавливается цвет элементов объекта ChartArea. With Charts { "Диаграмма" .ChartArea . Interior . ColorIndex = 3 . Border . ColorIndex = 5 End With
ChartTitle	Возвращает объект ChartTitle, являющийся заголовком диаграммы. В следующем примере задается заголовок диаграммы «Отчет»: With Charts ("Отчет") .HasTitle = True . ChartTitle . Text = "Продажи за май" End With
ChartType	Возвращает тип диаграммы: xlArea, xlBar, xlColumn, xlLine, xlPie, xlRadar, xlXYScatter, xlCombination, xl3DArea, xlSDBar, xl3DColumn, xl3DLine, xl3DPie, xlSDSurface, xlDoughnut
Legend	Возвращает объект Legend. Используется для изменения легенды. <i>Например:</i> ChartObjects(1).Chart.Legend.Font.Bold=True
PlotBy	Допустимые значения: xlColumns (столбцы используются как ряды данных при построении диаграммы), xlRows (строки используются как ряды данных при построении диаграммы)
DisplayBlanksAs	Допустимые значения: xlNotPlotted (при построении диаграммы пустые ячейки игнорируются); xlInterpoiated (значения в пустых ячейках интерполируются); xlZero (значения в пустых ячейках считаются равными нулю)

Для построения графика на пользовательской форме применяется элемент управления (рисунок), который используется для отображения графических файлов в формате .gif, .ipg, .ico. Свойства элемента управления Image, используемые для управления выводом изображения на пользовательскую форму, представлены в таблице 4.3.

Таблица 4.2 – Методы объекта ChartObjects

Метод	Значение
Add	Создает новую диаграмму. <i>Синтаксис: Add (Left, Top, Width, Height)</i> Left, Top – координаты левого верхнего угла диаграммы; Width, Height – ширина и высота диаграммы
Delete	Удаляет элемент семейства
CopyPicture	Копирует диаграмму в буфер обмена как рисунок. <i>Синтаксис: CopyPicture (Appearance, Format, Size)</i> Appearance – устанавливает способ копирования диаграммы. Значения: xlScreen (диаграмма копируется в том виде, в котором она отображается на экране) и xlPrinter (копируется так, как она выглядит при выводе на печать); Format – формат рисунка. Допустимые значения: xlPicture и xlBitmap
SetSourceData	Устанавливает источник данных диаграммы. <i>Синтаксис: SetSourceData (Source, PlotBy)</i> Source – диапазон, на основе которого строится диаграмма; PlotBy – допустимые значения: xlColumns (данные расположены по столбцам) и xlRows (данные расположены по строкам). <i>Пример: ActiveChart.SetSourceData Source :=Sheets (1) .Range ("A1 : F1"), PlotBy:=xlRows</i>
Export	Экспортирует диаграмму в графический формат. <i>Синтаксис: Export (FileName, FilterName, Interactive)</i> FileName – имя файла, в который будет записана диаграмма в графическом формате; FilterName – имя графического фильтра в том виде, как он записан в графическом формате; Interactive – допустимые значения: True (для того чтобы показать диалоговое окно в процессе фильтрации) и False (в противном случае) <i>Пример экспорта диаграммы в GIF-файл:</i> <i>ActiveChart . Export FileName : ="График . gif " , FilterName := "GIF"</i>
Location	Передвигает диаграмму на новое место. <i>Синтаксис: Location (Where, Name)</i> Where – указывает, будет ли диаграмма располагаться на новом листе диаграмм (xlLocationAsNewSheet), внедряться как объект на рабочий лист (xlLocationAsobject) или ее местоположение будет определяться автоматически (xlLocationAutomatic); Name – имя листа диаграммы, если аргумент where принимает значение xlLocationAsNewSheet, либо имя рабочего листа, если аргумент where принимает значение xlLocationAsobject
BringToFront	Отображает диаграмму на переднем плане
SendToBack	Отображает диаграмму на заднем плане
Select	Выбирает диаграмму

Таблица 4.3 – Свойства элемента управления Image

Свойство	Значение
AutoSize	<i>True</i> – рисунок автоматически изменяет размер для того, чтобы отобразить изображение целиком; <i>False</i> – в противном случае
Picture	Задаёт изображаемый графический файл. Используется совместно с функцией Loadpicture . <i>Синтаксис: Picture = LoadPicture(полное имя файла)</i> , т. е. полное имя отображаемого графического файла
SizeMode	Устанавливает масштабирование рисунка. fmPictureSizeModeClip – не помещающиеся в границах объекта части рисунка обрезаются. fmPictureSizeModeStretch – так, чтобы он занимал полную поверхность объекта. fmPictureSizeModeZoom – рисунок масштабируется с сохранением относительных размеров так, чтобы он помещался внутри объекта
PictureAlignment	Устанавливает расположение рисунка внутри объекта

4.2 Задания к лабораторной работе

Построить график функции на заданном диапазоне с заданным шагом на пользовательской форме согласно варианту из таблицы 4.4. Предусмотреть обработку ошибок.

Таблица 4.4 – Варианты исходных данных

Номер варианта	Функция	Отрезок $[a, b]$	Шаг h , град
1	$y = 2 \sin(x) \cos(x)$	$[0^\circ, 360^\circ]$	20
2	$y = 2 \sin^2(x) + \cos(x)$	$[0^\circ, 200^\circ]$	10
3	$y = \sqrt[3]{2 \sin(x) \cos(x)}$	$[-100^\circ, 100^\circ]$	20
4	$y = \sin(2x) \cos^3(x)$	$[30^\circ, 360^\circ]$	15
5	$y = \sqrt[3]{2 \sin(x) \cos(x^2)}$	$[-45^\circ, 90^\circ]$	15
6	$y = 2 \operatorname{tg}(x+1)$	$[0^\circ, 180^\circ]$	9
7	$y = -\sin^3(x) \sqrt{\cos(x)}$	$[-30^\circ, 120^\circ]$	15
8	$y = 2 \operatorname{tg}(\sqrt{x}) \cos(3x)$	$[0^\circ, 360^\circ]$	20
9	$y = x \sin(x)$	$[0^\circ, 200^\circ]$	10
10	$y = -x \cos(x)$	$[-100^\circ, 100^\circ]$	20
11	$y = \sin^2(x) + x$	$[30^\circ, 360^\circ]$	15
12	$y = \sqrt[3]{2 \sin(x) \cos(x)}$	$[-45^\circ, 90^\circ]$	15
13	$y = \operatorname{tg}^3(x) + \sqrt{\cos(x)}$	$[0^\circ, 180^\circ]$	9
14	$y = \sqrt{\sin^3(x) \cos(x^2)}$	$[-30^\circ, 120^\circ]$	15

4.3 Пример выполнения задания

Построить на листе Excel график функции $y = \frac{\sin x}{x^2 + 1}$ на отрезке $[a;b]$ с шагом h . Построение графика на листе Excel обычно производится через процедуру.

```
Public Sub primergrafik()
Dim a As Double, b As Double, h As Double
Worksheets(1).Range("A:B").Select
Selection.Clear
Worksheets(1).ChartObject.Delete
a = Cdbl(InputBox("Введите значение a"))
b = Cdbl(InputBox("Введите значение b"))
h = Cdbl(InputBox("Введите значение h"))
j = 1
For i = a To b Step h
Worksheets(1).Range("A" & j) = i
Work-
sheets(1).Range("B"&j)=Sin((i*3.14)/180)/(i^2+1
)
j = j + 1
Next i
график
End Sub

Public Sub график()
n=Application.CountA(Worksheets(1).Range("A
:A"))

Range("A1:B" & CStr(n)).Select

Charts.Add

ActiveChart.ChartType= xlXYScatterLinesNo-
Markers

ActiveChart.SetSourceData Source := Sheets
("Лист1 ").Range("A1:B" & CStr(n))
ActiveChart.LocationWhere:= xlLocationAsOb-
ject, Name:="Лист1"
End Sub
```

Выделение двух столбцов
первого листа
Очистка листа. Удаление
диаграммы
Ввод данных

В цикле выводятся данные
на лист Excel, в столбец A –
значения аргумента, B –
значения функции

Вызов процедуры график

Присваивание переменной
n количества заполненных
строк на листе в столбце A
Выбор данных для постро-
ения графика
Добавление листа диаграмм
в рабочую книгу

Выбор типа диаграммы
Установка источника дан-
ных диаграммы

Для построения графика на пользовательской форме создадим форму, на которой предусмотрим поля для ввода данных и область для вывода графика, а также кнопки для запуска построения графика, очистки формы и выхода (рисунок 4.1).

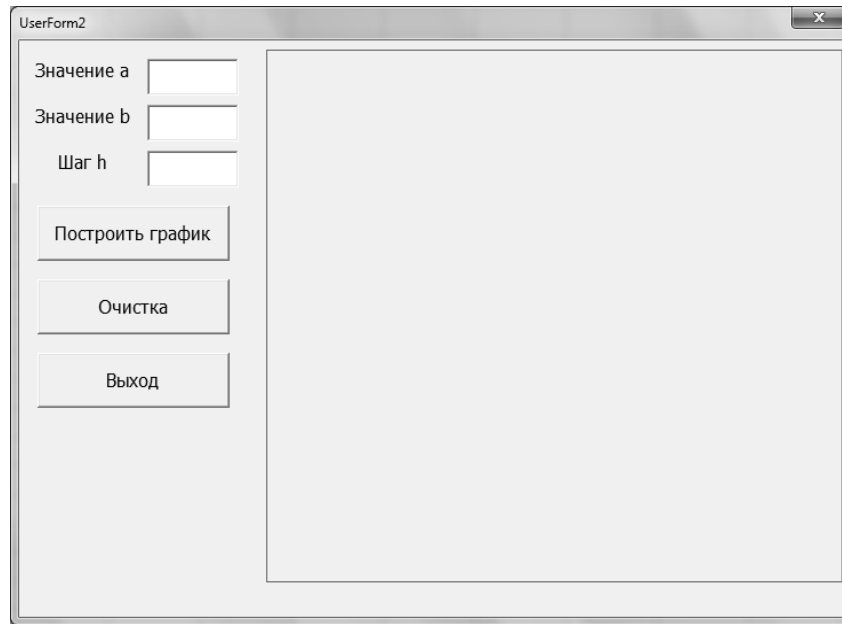


Рисунок 4.1 – Пользовательская форма

Текст программы представлен ниже.

```

Dim g As String
Private Sub CommandButton1_Click()
Dim a As Double, b As Double, h As Double
Worksheets(1).Range("A:B").Select
Selection.Clear
'Worksheets(1).ChartObject.Delete
a = Cdbl(TextBox1.Text)
b = Cdbl(TextBox2.Text)
h = Cdbl(TextBox3.Text)
j = 1
For i = a To b Step h
Worksheets(1).Range("A" & j) = i
Worksheets(1).Range("B" & j)=Sin((i*3.14)/180)/(i^2+1)
j = j + 1
Next i
график
Image1.Picture = LoadPicture(g)
End Sub
Public Sub график()
n = Application.CountA(Worksheets(1).Range("A:A"))
Range("A1:B" & CStr(n)).Select
Charts.Add
ActiveChart.ChartType = xlXYScatterLinesNoMarkers
ActiveChart.SetSourceData Source:=Sheets("Лист1").Range("A1:B" &
CStr(n))
ActiveChart.Location Where:=xlLocationAsObject, Name:="Лист 1"
ActiveChart.Export "d:\grafik.gif"
g = "d:\grafik.gif"
End Sub

```

```
Private Sub CommandButton2_Click()
Worksheets(1).Range("A:B").Select
Selection.Clear
'Worksheets(1).ChartObject.Delete
Image1.Picture = LoadPicture("")
End Sub
```

```
Private Sub CommandButton3_Click()
UserForm1.Hide
End Sub
```

```
Private Sub UserForm_Initialize()
Image1.PictureAlignment = fmPictureAlignmentTopLeft
Image1.PictureSizeMode = fmPictureSizeModeStretch
End Sub
```

Контрольные вопросы

- 1 Назначение элемента управления Image.
- 2 Свойства элемента управления Image.

5 Лабораторная работа № 16. Программирование на алгоритмическом языке

Цель работы: изучить основные графические методы VBA; получить навыки создания графических объектов.

5.1 Теоретические сведения

В VBA есть два графических объекта, которые позволяют работать с графикой: форма (UserForm) и управляющий элемент графическое поле (PictureBox). Все эти объекты способны содержать в себе точечный рисунок из графического файла, обладают графическими методами и позволяют с помощью графических методов рисовать на своей поверхности, а также способны содержать в себе другие управляющие элементы

Графические методы – это функции, которые содержатся в языке VBA и вызываются во время работы приложения. Графические методы ориентированы на *абсолютную систему координат* начало которой находится в верхнем левом углу экрана (рисунок 5.1).

Для изменения масштаба используется метод ***Scale***.

Синтаксис: [ИмяОбъекта].Scale(x1, y1) – (x2, y2),

где x1, y1 – координаты верхнего левого угла экрана;

x2, y2 – координаты правого нижнего угла экрана.

Если координаты опущены, то на рабочей поверхности объекта будет принята система координат по умолчанию (с единицей измерения – твип). Метод

Scale не изменяет размеры объекта, а задает значения координат его левого верхнего и нижнего правого углов. Например, чтобы в рисунке задать единицы измерения, соответствующие области изменения X от -10 до 10 , а Y от -20 до 20 , в программном коде можно записать такую команду: `Picture1.Scale (-10, 20) - (10, -20)`.

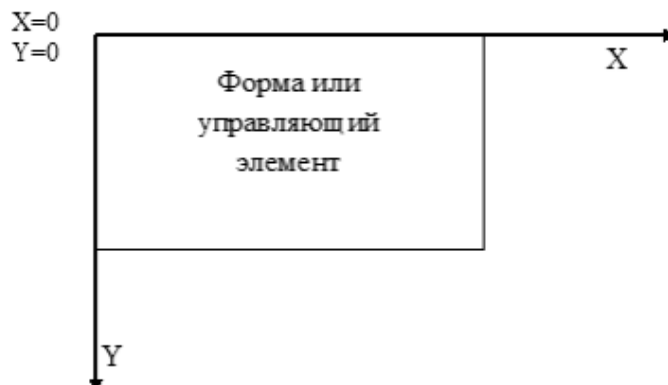


Рисунок 5.1 – Абсолютная система координат в VBA

Точка рисуется методом **Pset**.

Синтаксис: `[ИмяОбъекта].Pset(x,y)[,цвет]`,

где (x,y) – координаты точки;

цвет, по умолчанию, задается **черный**.

Отрезок и прямоугольник рисуется методом **Line**.

Синтаксис: `[ИмяОбъекта].Line[(x1,y1)-(x2,y2)[,цвет][,B[F]]]`,

где $(x1,y1)$ – координаты начальной точки отрезка. Если они не указаны, отрезок начинается от текущей позиции;

$(x2,y2)$ – координаты конечной точки отрезка.

Если указан параметр **B**, то вместо линии вычерчивается прямоугольник; пары координат при этом определяют положение его противоположных по диагонали углов.

Если указан параметр **F**, то прямоугольник закрашивается тем же цветом, что и границы.

Окружность, дуга, сектор, эллипс рисуются методом **Circle**.

Синтаксис: `[ИмяОбъекта].Circle(x,y), R [,Цвет] [[-]n1,[-]n2 [,n3]]`,

где (x,y) – координаты центра окружности (дуги, сектора, эллипса);

R – радиус;

$n1, n2$ – определяют углы начала и конца дуги в радианах и отсчитываются против часовой стрелки. Знак «минус» перед параметрами является признаком сектора, а при его отсутствии вычерчивается дуга;

$n3$ – определяет коэффициент сжатия при вычерчивании эллипса. Если его значение меньше 1, то осуществляется сжатие в вертикальном направлении, если больше 1 – в горизонтальном.

Рисование на листе Excel (Worksheet) в VBA осуществляется путём использования объектов векторной графики Micro Soft Office(MSO). Графические

фигуры, представленные на панели рисования MSO, образуют в VBA семейство объектов Shapes. Для рисования используются варианты метода добавления объектов этого семейства – **Add: AddLine** – добавить линию, **AddCurve** – добавить кривую, **AddPolyLine** – добавить полилинию, **AddShape** – добавить фигуру.

Рассмотрим методы, создающие новые элементы семейства **Shapes**.

Добавление отрезка прямой линии выполняется командой

```
ActiveDocument.Shapes.AddLine X1, Y1, X2, Y2,
```

где `ActiveDocument` – активный документ, в который добавляется линия, это имя можно опускать в специализированных модулях, предназначенных для работы с рабочими листами в Excel – Лист1 и т. п.;

`Shapes` – имя коллекции фигур;

`X1, Y1, X2, Y2` - горизонтальные и вертикальные координаты начальной и конечной точек отрезка прямой соответственно.

Метод **AddShape** добавляет объект Shape.

Синтаксис: `AddShape (Type, Left, Top, Width, Height)`,

где `Type` – тип объекта (либо целое число от 1 до 138 – номер фигуры в списке автофигур на панели рисования, либо название фигуры, например **msoShapeTriangle** (треугольник) **msoShapeRectangle** (прямоугольник), **msoShapeOval** (овал). Основные фигуры приведены в таблице 5.1);

`Left, Top` – координаты левого верхнего угла объекта;

`Width, Height` – ширина и высота объекта.

Например, вставить фигуру круг радиусом 4, центр которого находится в точке $A(x_0, y_0)$, можно двумя способами:

1) `ActiveSheet.Shapes.AddShape(msoShapeOval, x0, y0, 4, 4);`

2) `ActiveSheet.Shapes.AddShape 9, x0, y0, 4, 4.`

Таблица 5.1 – Типы объектов

Тип объекта	Номер фигуры	Объект
<code>msoShapeRectangle</code>	1	Прямоугольник
<code>msoShapeParallelogram</code>	2	Параллелограмм
<code>msoShapeTrapezoid</code>	3	Трапеция
<code>msoShapeOctagon</code>	6	Восьмиугольник
<code>msoShapeIsoscelesTriangle</code>	7	Равнобедренный треугольник
<code>msoShapeOval</code>	9	Эллипс

По умолчанию фигуры создаются окрашенными в черный цвет. Для изменения цвета фигуры следует воспользоваться свойством `ForeColor`:

`Shapes("ИмяФигуры").Line.ForeColor.RGB = НомерЦвета`

Наряду с именем фигуры допустимо использовать её номер.

Номер цвета может быть задан либо с помощью функции **RGB(r,g,b)**,

r – красный, g – зелёный, b – синий – беззнаковые целые типа байт. В таблице 5.2 приведены значения (r, g, b) для некоторых стандартных цветов либо через именованные константы, имена которых состоят из префикса vb и имени цвета: black – чёрный, blue – голубой, red – красный, yellow – жёлтый, white – белый и т. п.

Таблица 5.2 – Значения констант RGB

Цвет	Зачение Red	Значение Green	Значение Blue
Черный	0	0	0
Синий	0	0	255
Зеленый	0	255	0
Голубой	0	255	255
Красный	255	0	0
Фуксия	255	0	255
Желтый	255	255	0
Белый	255	255	255

Пример – Нарисовать прямую линию красного цвета, которая выходит из точки A(x_0, y_0) и заканчивается в точке B(x, y).

1-й вариант:

`ActiveSheet.Shapes.AddLine(x_0, y_0, x, y).Line.ForeColor.RGB = vbRed.`

2-й вариант:

`ActiveSheet.Shapes.AddLine(x_0, y_0, x, y).Line.ForeColor.RGB=RGB(255,0,0)`

Закрашивание созданных объектов осуществляется с помощью их свойства **Fill** – залить, при этом лицевой цвет заливки – **ForeColor** является её свойством, как и у линии.

Синтаксис: `Shapes(№ фигуры).Fill.ForeColor.RGB =НомерЦвета.`

Толщина линий определяется свойством **Weight**(вес) объекта **Shape**, определённого его номером или именем.

Синтаксис: `Shapes(№фигуры).Line.Weight = толщина.`

В этой команде слово «толщина» представляет или имя переменной типа **Single**, или непосредственно значение толщины в точках – **pt**.

Иногда построенные кривые требуют использовать для их наглядного восприятия кроме цвета различные стили рисования линий – пунктирный, штрих – пунктирный и т. д. Для этих целей предназначено свойство **DashStyle** метода **Line**, целые значения которого лежат между 1 и 8 (таблица 5.3).

Синтаксис: `Shapes(№фигуры).Line.DashStyle = №стиля.`

Аналогично стиль заливки задаётся методом **OneColorGradient** свойства **Fill**.

Синтаксис: `Shapes(№фигуры).Fill.OneColorGradient №стиля, ВариантСтиля, СтепеньСтиля.`

№стиля – принимает значения из таблицы 5.4. Со всеми типами градиентной заливки можно познакомиться на вкладке заливки фигур.

ВариантСтиля – определяет один из 4-х вариантов штриховки в способах заливки, кроме заливки от центра, для которой можно использовать только значения 1 и 2.

$0 \leq \text{СтепеньСтиля} \leq 1$, что эквивалентно выбору степени светлого.

Таблица 5.3 – Значения свойства DashStyle и стили линий









Значения № стиля	Имя константы (префикс: msoLine)	Стиль линии
1	Solid	
2	SquareDot	
3	RoundDot	
4	Dash	
5	DashDot	
6	DashDotDot	
7	LongDash	
8	LongDashDot	

Таблица 5.4 – Номер стиля заливки и вид штриховки

Значения № стиля	Имя константы (префикс: msoGradient)	Вид заливки
1	Horizontal	Горизонтальная
2	Vertical	Вертикальная
3	DiagonalUp	Диагональная 1
4	DiagonalDown	Диагональная 2
5	FromCorner	Из угла
6	FromTitle	От заголовка слайда
7	FromCenter	От центра

5.2 Задания к лабораторной работе

Вычертить на листе Excel схему механизма (рисунок 5.2) в заданном положении в соответствии с вариантом. Исходные данные для выполнения задания представлены в таблице 5.5.

5.3 Пример выполнения задания

Создадим из трех отрезков прямоугольный треугольник.

Текст программы:

```
Sub треугольник()
With ActiveDocument .Shapes
.AddLine 120, 100, 220, 100
.AddLine 120, 100, 170, 50
.AddLine 170, 50, 220, 100
.Range(Array(1, 2, 3)).Group
```

Рисуется прямая между точкой (120, 100) и точкой (220, 100)
 Прямая – между (120, 100) и (170, 50)
 Прямая – между (170, 50) и (220, 100)
 Группировка отрезков в одну фигуру

End With
End Sub

Здесь для уменьшения текста программы использован оператор **With ... End With**, присоединяющий имена методов добавления линии AddLine и группировки фигур 1, 2, 3, составляющих область: .Range(Array(1, 2, 3)).Group.

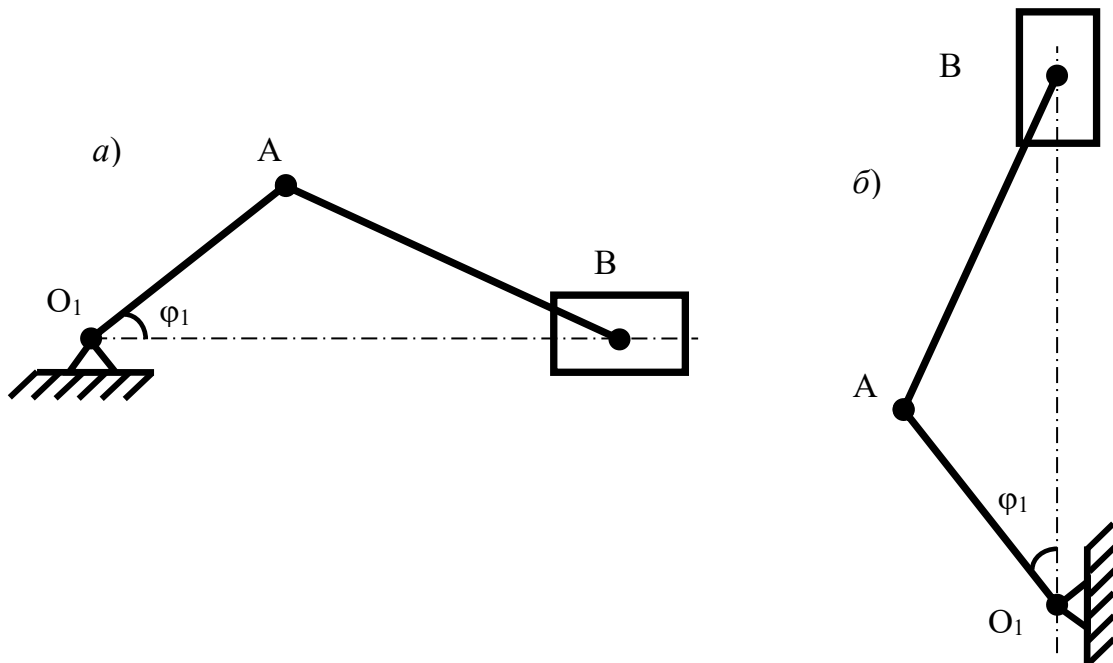


Рисунок 5.2 – Схемы механизма

Таблица 5.5 – Исходные данные

Вариант	Схема (см. рисунок 5.2)	Длина кривошипа l_{O_1A} , м	Длина шатуна l_{AB} , м	Угол поворота φ_1 , град
1	а	0,1	1,1 l_{O_1A}	20
2	б	0,15	1,2 l_{O_1A}	40
3	а	0,2	1,3 l_{O_1A}	60
4	б	0,25	1,4 l_{O_1A}	80
5	а	0,3	1,5 l_{O_1A}	100
6	б	0,35	1,6 l_{O_1A}	120
7	а	0,4	1,7 l_{O_1A}	140
8	б	0,45	1,8 l_{O_1A}	160
9	а	0,5	1,9 l_{O_1A}	180
10	б	0,55	1,1 l_{O_1A}	200
11	а	0,6	1,2 l_{O_1A}	220
12	б	0,65	1,3 l_{O_1A}	240
13	а	0,7	1,4 l_{O_1A}	260
14	б	0,75	1,5 l_{O_1A}	280

Контрольные вопросы

- 1 Команда построения отрезка.
- 2 Команда построения окружности.
- 3 Команда изменения веса линий.
- 4 Команда изменения типа линий.

Список литературы

- 1 **Гуриков, С. Р.** Информатика : учебник / С. Р. Гуриков. – 2-е изд., перераб. и доп. – Москва : ИНФРА-М ; ФОРУМ, 2020. – 630 с.
- 2 **Гвоздева, В. А.** Информатика, автоматизированные информационные технологии и системы: учебник / В. А. Гвоздева. – Москва: ФОРУМ; ИНФРА-М, 2021. – 542 с.
- 3 **Безручко, В. Т.** Информатика. Курс лекций : учебное пособие / В. Т. Безручко. – Москва : ФОРУМ ; ИНФРА-М, 2020. – 432 с.
- 4 **Баранова, Е. К.** Основы информатики и защиты информации: учебное пособие / Е. К. Баранова. – Москва: РИОР; ИНФРА-М, 2018. – 183 с.
- 5 **Кильдишов, В. Д.** Использование приложения MS Excel для моделирования различных задач: практическое руководство / В. Д. Кильдишов. – Москва: СОЛОН-Пресс, 2015. – 156 с.
- 6 Подготовка и редактирование документов в MS WORD: учебное пособие / Е. А. Баранова [и др.]. – Москва: КУРС ; ИНФРА-М, 2017. – 184 с.