

/ " %) ' % &
) + 7 4 ! 3 H 7 ; ! 4 £ 3 9 7
 + 7 4 ! 3 " ")) & ! 3 ()
 2 7 ; ! 4 ! 4 (| & # | 6
) ! (' % % ' ()
)) 3 9 7 " # / " % , ' % &) +
 3 9 ! 3 H 3 7 ! 4 £ ? 7 7
 %+ 3 9 C ! 3 " ' ! ! ! & !) ! 3
 (2 3 7 C ! 4 ! 4 (| & # \$) !
 (' % % ' (, # A ! 3 ! 4 ' & ' % () B J ! ()
) % % # # ! 3 3 7 ! 4 3 7 E % ! F #
 5 " & % (" ' % &) +
 KL DG 4 ! 3 H : ! 4 E , F E 3 F
 " ! + 7 8 ! 3 H 7 : ! 4 £ 3 = 7 E 4 F
 7 4 ! 3 H 7 ; ! 4 £ 3 9 7 E 8 F
 3 9 ! 3 H ! 4 £ ? 7 7 E : F
 ! 3 3 7 ! 4 3 7 E : F

N2- O N = 0 + 2 - 2 - P + ; 2 ++ ; 2 / 7 / + 2 0
 I) - & (. " (M 4 + M : , " !
 " (" (E 4 F N E ; F # A A (, ' " % & # |
 / \$ 0 % ' , \$) 1 (* " % & # |
 1 ' .. (0 O P O 9 + 0 0 9 6 " ()
 " " ! ! # | / " " E 4 F N E ; F #
 I % . (0) % %
 " (M 4 , " # |
 " (& (# | - & ()
 & " % " ' & " (R 4 # S % / #
 \$ % & 2 . # T ()
 % " E # 4 F #

УДК 004
ББК 32.973-01
М 54

Рекомендовано к изданию решением
Учебно–методического совета
(протокол № 7 от 26.03.2010. г.) и
методическим советом факультета
(протокол № 5 от 25.02.2009 г.)
Ысык-Кульского государственного
университета им. К.Тыныстанова

Рецензент: . . к.ф.м.н. Тултуков Б.Т.

Составители: Бообеков Т.И., Балан Р.К., Асанова З.А.

М 54 Методическое указание к выполнению лабораторных работ по дисциплине «Язык программирования Visual Basic 6.0» / Сост.: Бообеков Т.И., Балан Р.К., Асанова З.А. / ЫГУ им. К. Тыныстанова. – Каракол, 2010. - 60стр.

ISBN 978-9967-431-80-5

Методическое пособие содержит краткие теоретические сведения, задания, примеры и рекомендации для выполнения лабораторных работ на языке Visual Basic 6.0. Предназначено для студентов специальностей «Программное обеспечение вычислительной техники и автоматизированных систем», «Автоматизированные системы обработки информации и управления» всех форм обучения.

М 2404090000-09
ISBN 978-9967-431-80-5

УДК 004
ББК 32.973-01

© Бообеков Т.И., Балан Р.К., Асанова З.А.
2010.
@ ЫГУ им. К.Тыныстанова, 2010.

Введение

Методическое указание предназначено для студентов при изучении курса «Язык программирования Visual Basic». Тематическое планирование учебного материала проведено в соответствии с действующей программой.

Цель курса «Язык программирования Visual Basic» состоит в том, чтобы познакомить студентов с основными концепциями программирования на языке высокого уровня. На протяжении курса студенты изучают такие базовые приемы программирования, как написание кода, создание форм, объявление переменных, вычисление выражений, использование ветвлений и циклических конструкций и др.

Все приведенные в курсе задания по разработке приложений ориентированы на использование Visual Basic.

В начале каждого лабораторного занятия кратко приведен теоретический материал, необходимый для решения некоторых задач по данной теме. После него предлагается решение некоторых задач и список задач для самостоятельного выполнения.

Фактический материал пособия неоднократно использовался авторами при проведении практических занятий по курсу «Язык программирования Visual Basic».

Лабораторная работа №1

Работа в среде Visual Basic

Цель занятия

Приобрести начальные навыки работы в среде Visual Basic. Изучить элементы среды программирования, порядок установки элементов на форму и управления размещением элементов.

Краткие теоретические сведения

Рабочее окно (рис. 1.1) представляет собой интегрированную среду разработки – интерфейс языка программирования Visual Basic. Эта среда может настраиваться с помощью диалогового окна, вызываемого командой Tools\Options. Рабочее окно предлагает целый набор инструментальных средств, которые можно использовать при создании программ.

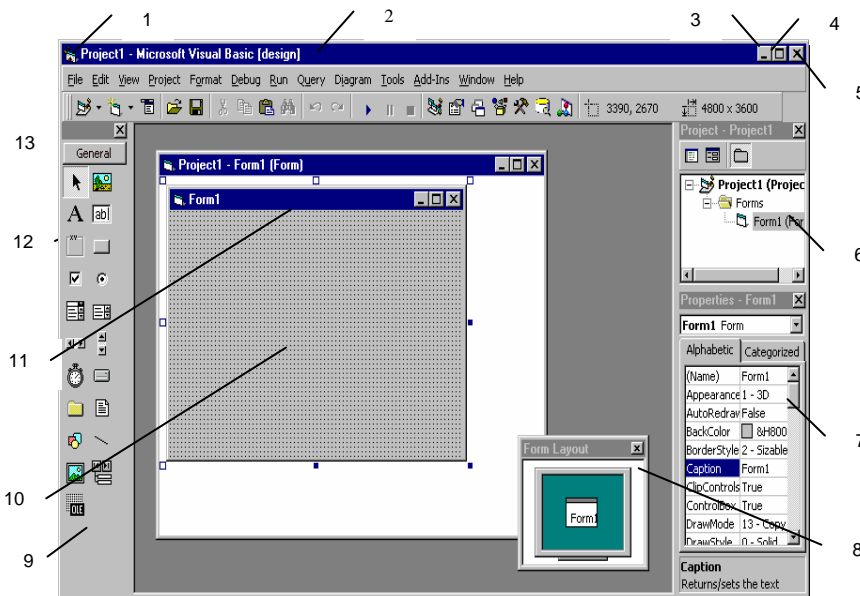


Рис. 1.1. Окно программы Visual Basic

1 – кнопка системного меню, 2 – заголовок, 3 – кнопка свертывания окна в пиктограмму, 4 – кнопка развертывания окна, 5 – кнопка закрытия окна, 6 – окно проекта, 7 – окно свойств, 8 – окно позиционирования формы, 9 – панель элементов управления, 10 – шаблон формы, 11 – конструктор форм, 12 –

Большинство элементов рабочего окна, характерны для приложений Windows: (заголовок, меню, панели инструментов). Интегрированная среда разработки проекта многооконная. Состав окон приведен на рис. 1.1.

Меню (Menu)

Строка Меню содержит список команд, предназначенных для управления разработкой проекта, пункты меню могут иметь несколько уровней вложения:

File (Файл) - содержит команды для работы с файлами создаваемых приложений, загрузки, сохранения, вывода на печать.

Edit (Правка) - содержит команды редактирования, предназначенные для создания исходного текста программы, включая средства поиска и замены.

View (Просмотр) - обеспечивает доступ к различным частям приложения и среды разработки VB.

Project (Проект) - предназначен для добавления новых объектов VB к разрабатываемым проектам, добавления или удаления элементов управления на панель элементов управления, настройки свойств проекта.

Format (Формат) – дает доступ к различным настройкам элементов управления, размещенных на создаваемых программистом формах.

Debug (Отладка) – содержит средства, предназначенные для отладки программ или поиска ошибок.

Run (Выполнение) – служит для запуска и остановки программ непосредственно из среды разработки.

Tools (Инструменты) – обеспечивает доступ к работе с процедурами и меню внутри приложения. Этот пункт меню имеет важную команду Options, которая открывает одноименную диалоговую панель с закладками Options, где настраивается практически вся среда разработки Visual Basic.

Add-Ins (Добавить в ...) – дает доступ к инструментам, которые могут быть добавлены к окружению VB: мастера, ActiveX – элементы и другое.

Diagram (Диаграммы) – содержит средства для оформления диаграмм.

Window (Окно) – используется для работы с окнами в среде разработки.

Query – доступ к внешним базам данных.

Help – справочная система.

Выбор пунктов меню осуществляется мышью или клавишами. При управлении с помощью клавиатуры для входа в меню используется клавиша Alt. Выбор пунктов меню осуществляется с помощью клавиш управления курсором или с использованием горячих клавиш (подчеркнутые символы в командах меню): [Alt- клавиша].

Панели инструментов (Toolbars)

VB имеет четыре стандартные панели инструментов: Standard – стандартная, Edit - редактирования, Debug – отладки и Form Editor – редактор форм.

Стандартная панель инструментов содержит команды, дублирующие основные команды меню.

Панель редактирования используется при вводе и редактировании текста программы. Содержит кнопки для вывода всплывающих списков свойств и методов объекта, констант, синтаксиса для процедур и методов, а также содержит кнопки для управления редактированием текста.

Панель отладки – служит для отладки программ в процессе ее выполнения и обеспечивает запуск программы на выполнение, временную остановку (пауза), выход из программы, установку точек останова,

пошаговое выполнение в режиме пошагового выполнения, вычисление выделенного выражения.

Панель редактора форм – применяется при разработке форм. Она позволяет изменять порядок элементов управления, выравнивать их по горизонтали и вертикали, уравнивать размеры выделенных элементов управления по ширине и высоте, а также блокировать или разблокировать элементы управления в форме.

Панель инструментов элементов управления и компонентов пользователя (Toolbox)

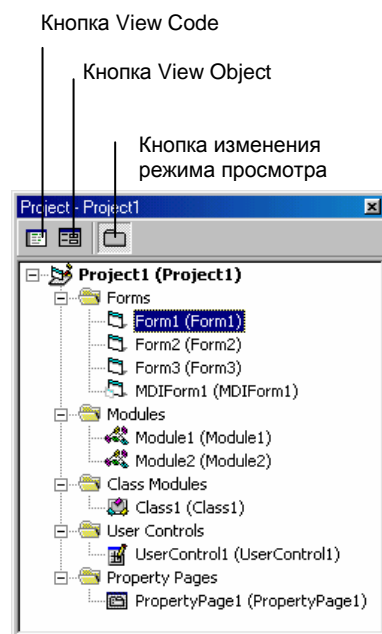


Рис.1.2. Окно Проект

Панель Toolbox содержит элементы управления. Элементы управления – это элементы, которые используются при разработке интерфейса создаваемых приложений. Общее количество доступных к использованию элементов управления зависит от того, какая версия VB используется.

Для добавления новых компонентов к панели инструментов Toolbox из числа зарегистрированных необходимо:

- ввести команду Project\Components, выбрать закладку Controls;
- найти в списке нужный элемент управления и установить напротив него флажок;
- выйти из окна диалога, щелкнув кнопку Ok.

Форма (Form)

Создаваемые в Visual Basic окна называются формами. Форма – это основное окно интерфейса разрабатываемой программы, форма - это также основа для создания окон диалога.

Окно Проект (Project)

В окне проекта (броузер проектов) (рис. 1.2.) отображаются все элементы приложения: формы, модули, классы и т.п., сгруппированные по категориям. В VB все разрабатываемые приложения называются проектами. Проект представляет группу связанных файлов и может содержать несколько групп компонентов (формы, модули и т.д.). Все проекты VB строятся по модульному принципу, поэтому и текст программы состоит не из одного большого файла, а из нескольких частей - процедур. Несколько проектов также могут объединяться в группы. Ниже заголовка окна проекта размещены три кнопки: кнопка просмотра текста

программы; кнопка просмотра объекта; кнопка изменения режима просмотра: в виде списка компонентов или в виде дерева групп компонентов. В качестве объектов могут быть формы, MDI-формы, модули, классы, управляющие элементы, страницы свойств. Модули и классы не имеют визуальных компонентов.

Окно Свойства (Properties)

В окне свойств (рис. 1.3.) задаются свойства выбранного элемента управления. В строке заголовка окна свойств рядом с текстом Properties указывается имя формы, которой принадлежит элемент управления. Поле со списком под строкой заголовка позволяет выбрать требуемый элемент управления. В списке, расположенном ниже, перечислены свойства этого элемента. Эти свойства могут быть упорядочены в алфавитном порядке (Alphabetic) или расположены по категориям (Categorized). Набор свойств зависит от типа элемента управления.

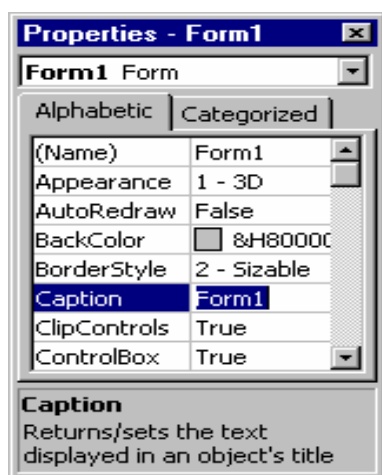


Рис. 1.3. Окно Свойства

Как установить свойства объекта? Имеется три способа:

1. ввести значение в поле справа от свойства;
2. выбрать из списка, который открывается щелчком мыши по полю;
3. установить с помощью окна диалога, при щелчке мышью по полю появляется кнопка (...) – троеточие или эллипсис. При щелчке по кнопке троеточие появляется окно диалога для настройки соответствующего свойства.

Окно Программа (Code)

Сразу после запуска окно Программа не отображается. Текст программы в VB разделяется на части – подпрограммы и записывается в процедуры обработки событий или процедуры пользователя. Поэтому текст программы, как правило, связан с определенным элементом управления. Это позволяет открыть окно диалога двойным щелчком по соответствующему элементу формы или по самой форме.

Кроме того, окно программы можно открыть щелчком по кнопке View Code в окне Project.

В верхней строке окна программы (рис.1.4) располагается строка заголовка, в которой указано имя проекта и управляющие кнопки (системного меню, закрытия окна, свертывания и развертывания окна).

Ниже строки заголовка расположены два раскрывающихся списка: список объектов (левый) и список процедур (правый).

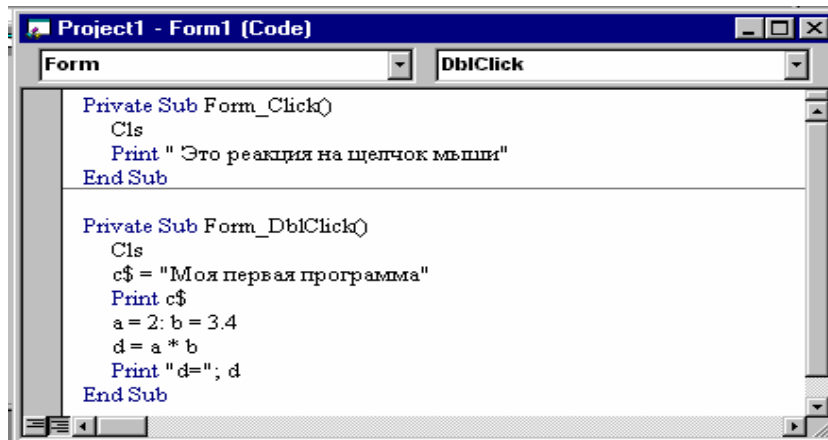


Рис. 1.4. Окно Программы (Code)

Список объектов содержит все объекты текущей формы. В их число входит и специальный объект General, содержащий общий код, используемый всеми процедурами формы.

Список процедур содержит список всех событий, распознаваемых текущим объектом. Если для объекта уже написаны процедуры обработки событий, то они выделяются здесь жирным шрифтом. Если выбрать любой пункт данного списка, то VB выведет соответствующую процедуру либо ее шаблон в окне программы и поместит в нее курсор

Окно позиционирования формы (Form Layout)

Данное окно позволяет установить место, где будет размещаться форма при запуске программы.

Задание

1. Изучите среду разработки проекта.
2. Изучите приемы установки элементов управления на форму и управления размещением элементов управления на форме.

Порядок выполнения задания

1. Запустите программу Visual Basic: введите команду Пуск\Программы, найдите в подменю команды Программы пиктограмму Visual Basic 6.0 и щелкните по ней дважды мышью. Если данной программы нет в меню, найдите ее в компьютере командой и Пуск\Найти\Файлы и папки. Имя исполняемого файла – Vb6.exe (могут быть и другие способы запуска Vb6 в зависимости от опыта пользователя и установленного программного обеспечения).

2. Щелкните в окне диалога New Project дважды по пиктограмме Standard.EXE для создания стандартного приложения Windows (это окно

может не появляться на экране, а сразу появится рабочее окно Visual Basic).

3. Изучите элементы рабочего окна VB (среда разработки программы) и команды меню. Измените размеры и положение формы в окне, перетаскивая его мышью.

4. Закройте окна Проект и Свойства кнопками закрытия окон.

5. Откройте окна Проект и Свойства командами View\Project Explorer и View\Properties Window, соответственно. Найдите в стандартной панели инструментов одноименные кнопки.

6. Переместите окно Проект в нижний правый угол рабочего окна. Переместите окно Проект в первоначальное положение.

7. Закройте панель элементов управления (Toolbox). Выведите панель Toolbox в рабочее окно командой View\Toolbox.

8. Откройте новую форму. Присвойте имя, наименование, установите размеры формы и ее положение в проекте.

– Запустите программу командой Run, Start. Появится пустая форма.

– Закройте проект. Установите положение формы в проекте с помощью окна Form Layout. Снова запустите проект и проверьте положение формы в окне проекта.

9. Установите на форму пять меток и пять окон ввода (Label, Text Box). Опишите в таблице их свойства и присвойте значения этих свойств элементам управления:

Тип объекта	Имя (Name)	Наименование (Caption)	Положение верхнего левого угла		Размеры	
			сверху (Top)	слева (Left)	ширина (Width)	высота (Height)
Метка (Label)						
Окно ввода (Text Box)						

Можно также добавить такие свойства как цвет, шрифт и т.п.

10. Упорядочите элементы управления на форме с помощью опций команды главного меню Format.

11. Установите на форму рамку (Frame). Скопируйте в нее несколько элементов управления с формы, используя контекстное меню, и выровняйте их по вертикали.

12. Добавьте на панель элементов управления сетку Microsoft Flex Grid. Введите команду Project\Components\Controls, найдите в списке объект Microsoft FlexGrid Control 6.0 (SP3) и щелчком мыши установите напротив него флажок. Щелкните по кнопке Ok.

13. Добавьте к проекту еще одну форму командой Project\Add Form.
14. Создайте новую папку на рабочем диске и сохраните проект в этой папке командой File\Save As.
15. Откройте новый проект командой File\New Project.
16. Откройте сохраненный ранее проект командой File\Open Project.
17. Выведите на печать форму Form1. Введите команду File\Print, установите флажок Form Image и щелкните по кнопке Ok – будет напечатана форма со всеми установленными на ней элементами (или пустая форма).
18. Выйдите из программы Visual Basic.

Указание к выполнению задания

Выполните последовательно все пункты задания. По ходу выполнения задания оформите отчет, отражая в нем порядок выполнения задания, вводимые команды и результаты выполнения команд. Вводимые команды описывать, например, следующим образом: Файл\Сохранить, указать диск, маршрут и имя файла.

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, описание порядка выполнения пунктов задания, ответы на контрольные вопросы.

Контрольные вопросы:

1. Как запустить программу VB?
2. Назовите основные элементы рабочего окна VB.
3. Как установить элементы управления на форму?
4. Какая команда используется для управления размещением элементов управления на форме?
5. Как добавить новые элементы управления на панель инструментов Toolbox?
6. Для чего предназначено окно Проект?
7. Каково назначение окна Свойства?
8. Для чего предназначено окно Программа?
9. Как вызвать окно Программа?
10. Где записывается текст программы объекта?
11. Как сохранить программу на диске?
12. Как загрузить существующую программу?
13. Как вывести на печать текст программы?
14. Как вывести на печать сведения об установленных свойствах формы?

Лабораторная работа №2

Разработка программ в среде Visual Basic (окно программы)

Цель работы:

Ознакомиться со средой программирования и приобрести первоначальные навыки в разработке программ в среде Visual Basic. Ознакомиться с операторами Let и Print.

Краткие теоретические сведения

Выражения представляют собой комбинацию переменных и операндов, соединенных знаками отношения. В зависимости от используемых знаков операций различают арифметические выражения, логические выражения и строковые выражения.

Синтаксис всех выражений одинаков:

<операнд> знак_операции <операнд>

В качестве операндов в выражениях могут использоваться константы, переменные, функции и другие выражения. В табл. 2.1 приведены основные математические функции, используемые в языке Visual Basic. Они во многом подобны функциям языка программирования Qbasic.

Для вычисления логарифма числа по произвольному основанию и для вычисления обратных тригонометрических функций полезно запомнить следующие формулы преобразования:

$$\text{Log}_a(x) = \frac{\text{Log}_b(x)}{\text{Log}_b(a)} \quad (2.1)$$

$$\arcsin(x) = \arctg(x / \sqrt{1-x^2}) \quad (2.2)$$

$$\arccos(x) = \pi/2 - \arctg(x / \sqrt{1-x^2}) \quad (2.3)$$

$$\text{arcctg}(x) = \pi/2 - \arctg(x) \quad (2.4)$$

Операторы ввода и вывода данных

Оператор Let. Служит для присвоения значений переменным:

Let <имя переменной> = <выражение>

Let a=2

Let b1=Sin(x)+exp(x)

Оператор LET может быть опущен, поэтому выражения

Let A= Sqr(x)+ Log(x) и A= Sqr(x)+ Log(x)

эквивалентны.

Метод Print выводит данные на форму. Синтаксис метода Print:

Print "Комментарий" [;/ /,] <список выражений> [;/,]

Комментарии делают понятным пользователю, что выводится на экран методом Print. В качестве выражений могут использоваться константы, переменные, функции и выражения:

Print x, y

Print "x=";x, "y=";y

Print "Значение функции Sin(x) при x = ";x;" равно ";Sin(x)

При использовании в качестве разделителя символа ";" значения выводятся без пробела, при использовании в качестве разделителя символа "," значения выводятся в разных зонах.

Таблица 2.1

Математические функции

Функция	Описание
Abs(N) ¹	Вычисляет абсолютное значение числа N
Atn(N)	Вычисляет арктангенс числа N. Значение аргумента должно находиться в интервале от -1 до +1. ²
Cos(N)	Вычисляет косинус аргумента.
Exp(N)	Возвращает число e, возведенное в указанную степень
Fix(N)	Возвращает целое число, меньшее или равное N для положительных чисел и большее или равное N для отрицательных чисел.
Int(N)	Возвращает целое число, меньшее или равное N как для положительных, так и для отрицательных чисел
If(усл., V1,V2)	Условная функция, возвращает результат согласно выражению V1, если условие истинно, и согласно выражению V2, если условие ложно.
Log(N)	Вычисляет натуральный логарифм аргумента.
Rnd(N)	Возвращает случайное число в интервале от 0 до 1. При N<0 возвращает определенное число, зависящее от N; при N=0 – псевдослучайное число; при N>0 – новое случайное число.
Round(N[, дес])	Возвращает число, округленное к заданному числу десятичных знаков.
Sgn(N)	Возвращает знак числа: 1 , если N>0; 0 , если N=0; -1, если N<0.
Sin(N)	Вычисляет синус угла.
Sqr(N)	Вычисляет корень квадратный из аргумента.
Tan(N)	Вычисляет тангенс угла.
Val(C)	Преобразует аргумент строкового типа в число.

¹ N – переменная числового типа, C – строковая переменная

В методе Print могут использоваться функции Tab и Spc:

```
Print "x=";x; Tab(20); "y=";y
```

Результат:

```
1.754      3,564354
```

```
Print "x=";x; Spc(20); "y=";y
```

Результат:

```
1.754      3,564354
```

При использовании функции Tab отсчет символов ведется от начала экрана, а при использовании функции Spc – от текущего положения курсора.

Если в конце оператора Print стоят символы разделители, то курсор ввода остается в текущей строке, и следующий оператор Print будет выводить информации с этой позиции. Если символы-разделители в конце оператора Print отсутствуют, то курсор переводится на следующую строку.

Задание

Протабулировать функцию одной переменной на заданном отрезке. Вычислить сумму и произведение значений функции на отрезке табулирования. Найти глобальные экстремумы функции на отрезке табулирования.

Указания к выполнению задания

Текст программы поместить в обработчик события Click формы.

Значения переменных a и b, начального и конечного значений аргумента, шага табулирования вводить с помощью оператора Let. Вывод данных на форму осуществлять с помощью оператора Print. Для управления выводом результатов на экран использовать в операторе Print управляющие операторы ";", ",", а также функции Tab и Spc. Для ограничения числа десятичных знаков в данных, выводимых на экран, использовать функцию Round. Границы отрезка табулирования выбрать самостоятельно в соответствии с областью определения функции.

2 В тригонометрических функциях аргумент измеряется в радианах. Для пересчета радианов в градусы надо умножить значение функции на $180/\pi$. Для пересчета градусов в радианы надо умножить значение функции на $\pi/180$. Приближенное значение числа π равно 3,141593, а более точное значение можно вычислить по формуле $\pi = 4 * \text{Atn}(1)$

Варианты заданий

1.	$y = \begin{cases} ax^3 + b \ln 2x , \sqrt{a-b} < x \\ \sqrt{a + \sin(2x)} - e^{3 x }, \sqrt{a-b} > x \end{cases}$	14.	$y = \begin{cases} \ln(x^2) - e^x + \lg a-b , 2a+b < 0 \\ \arctg(2x-0.5) + \sqrt{a+bx}, 2a+b \geq 0 \end{cases}$
2.	$y = \begin{cases} e^{\sin(x)} + b\sqrt{2\cos(6x-0.3)}, b^2 > ax \\ e^{-x} + \sqrt{\lg 3x+0.6 }, b^2 \leq ax \end{cases}$	15.	$y = \begin{cases} a^x - e^x + b^3 \cos(4x-0.2), a^2 - b^2 < 10x \\ \lg(4.5x) + \frac{x}{\sin(0.5x)}, a^2 - b^2 \geq 10x \end{cases}$
3.	$y = \begin{cases} a\sqrt{ \sin(x) + \cos^2(x)} + e^{bx}, \sqrt{bx^3+6} < a \\ (x^{-3} - b)\cos(3x-0.5), \sqrt{bx^3+6} \geq a \end{cases}$	16.	$y = \begin{cases} \ln \left \frac{x^{-1}}{ax+b} \right - e^{2x}, \sin(x) < \sqrt{a^2+b} \\ x^3 + 2ax - \sqrt{b^2} - \lg(x) , \sin(x) \geq \sqrt{a^2+b} \end{cases}$
4.	$y = \begin{cases} \ln(x^2) - e^{\frac{ax}{b}} + \lg a-b , 2a-b < 0 \\ \arctg(2x+0.5) + \sqrt{a+bx}, 2a-b \geq 0 \end{cases}$	17.	$y = \begin{cases} e^{2x} + b\sqrt{a^4 + bx^2}, b < a^{-2} + x \\ \sqrt{ ax^4 + 4bx^2 }, b \geq a^{-2} + x \\ \lg(a+x) \end{cases}$
5.	$y = \begin{cases} \frac{bx^3}{a+bx} + \sin^2(a), ba < 2x \\ (ax^3 + 6x^2 + 105)\sqrt[3]{\sin(ba)}, ba \geq 2x \end{cases}$	18.	$y = \begin{cases} \frac{a^3x + \sqrt{a^2 + 7 bx }}{2ab \cos(2x)}, \sqrt{ x } \geq 2ab \\ \lg \left \frac{bx^3}{ax+1} \right , \sqrt{ x } < 2ab \end{cases}$
6.	$y = \begin{cases} bx^3 - a \ln 2x + \sin^2(x), \sqrt{(b-a)} < x \\ \sqrt{ b + \lg(x)} - e^{3x}, \sqrt{(b-a)} \geq x \end{cases}$	19.	$y = \begin{cases} \frac{2}{3}a - b^2 \sin\left(\frac{x}{2} - 0.6\right), a^2 + b^2 < 3x \\ a \cos(x^2 - b) + \sqrt[3]{b^2}, a^2 + b^2 \geq 3x \end{cases}$
7.	$y = \begin{cases} ax^3 + b \ln 2x , \sqrt{a-b} < x \\ \sqrt{a + \sin(2x)} - e^{3 x }, \sqrt{a-b} > x \end{cases}$	20.	$y = \begin{cases} \ln(x^2) - e^x + \lg a-b , 2a+b < 0 \\ \arctg(2x-0.5) + \sqrt{a+bx}, 2a+b \geq 0 \end{cases}$
8.	$y = \begin{cases} e^{\sin(x)} + b\sqrt{2\cos(6x-0.3)}, b^2 > ax \\ e^{-x} + \sqrt{\lg 3x+0.6 }, b^2 \leq ax \end{cases}$	21.	$y = \begin{cases} a^x - e^x + b^3 \cos(4x-0.2), a^2 - b^2 < 10x \\ \lg(4.5x) + \frac{x}{\sin(0.5x)}, a^2 - b^2 \geq 10x \end{cases}$
9.	$y = \begin{cases} a\sqrt{ \sin(x) + \cos^2(x)} + e^{bx}, \sqrt{bx^3+6} < a \\ (x^{-3} - b)\cos(3x-0.5), \sqrt{bx^3+6} \geq a \end{cases}$	22.	$y = \begin{cases} \ln \left \frac{x^{-1}}{ax+b} \right - e^{2x}, \sin(x) < \sqrt{a^2+b} \\ x^3 + 2ax - \sqrt{b^2} - \lg(x) , \sin(x) \geq \sqrt{a^2+b} \end{cases}$
10.	$y = \begin{cases} \ln(x^2) - e^{\frac{ax}{b}} + \lg a-b , 2a-b < 0 \\ \arctg(2x+0.5) + \sqrt{a+bx}, 2a-b \geq 0 \end{cases}$	23.	$y = \begin{cases} e^{2x} + b\sqrt{a^4 + bx^2}, b < a^{-2} + x \\ \sqrt{ ax^4 + 4bx^2 }, b \geq a^{-2} + x \\ \lg(a+x) \end{cases}$
11.	$y = \begin{cases} \frac{bx^3}{a+bx} + \sin^2(a), ba < 2x \\ (ax^3 + 6x^2 + 105)\sqrt[3]{\sin(ba)}, ba \geq 2x \end{cases}$	24.	$y = \begin{cases} \frac{a^3x + \sqrt{a^2 + 7 bx }}{2ab \cos(2x)}, \sqrt{ x } \geq 2ab \\ \lg \left \frac{bx^3}{ax+1} \right , \sqrt{ x } < 2ab \end{cases}$

12.	$y = \begin{cases} bx^3 - a \ln 2x + \sin^2(x), \sqrt{(b-a)} < x \\ \sqrt{ b + \operatorname{tg}(x) - e^{3x} }, \sqrt{(b-a)} \geq x \end{cases}$	25.	$y = \begin{cases} \frac{2}{3}a - b^2 \sin\left(\frac{x}{2} - 0.6\right), a^2 + b^2 < 3x \\ a \cos(x^2 - b) + \sqrt[3]{b^2}, a^2 + b^2 \geq 3x \end{cases}$
13.	$y = \begin{cases} \operatorname{arctg}(5x - 0.7a), a + 2b \geq 0 \\ \ln a + b + \sin(ax), a + 2b < 0 \end{cases}$	26.	$y = \begin{cases} (x^2 - a) \cos(x) - b\sqrt{2}, a + x^2 \geq b \\ b^{-2} - \sqrt{\operatorname{tg}(2x - 1)}, a + x^2 < b \end{cases}$

Пример программы

```
Private Sub Form_Click()
```

```
a = 12: b = 6: h = 2.5
```

```
s = 2 * (a + b) * h
```

```
x=5
```

```
if x<a then
```

```
  y=sin(x)
```

```
else
```

```
  if x<=0 then Print "нет решения": exit sub
```

```
  y= log(x)
```

```
end if
```

```
Print x, Round(y,2)
```

```
End Sub
```

Требования к оформлению отчета:

Отчет должен содержать тему, цель занятия, задание, текст программы и схему алгоритма.

Контрольные вопросы

1. Приведите формат оператора Let и дайте пояснение.
2. Приведите формат оператора Print и дайте пояснение.
3. Поясните принцип работы программы по схеме алгоритма.

Лабораторная работа №3. Создание простой формы

Цель занятия: закрепить знания, полученные на первых занятиях.

Дополнительные сведения о вводе и выводе данных

В лабораторной работе № 2 для ввода и вывода данных использовались операторы Let и Print, с которыми Вы знакомы по языку QBasic. В языке Visual Basic для ввода и вывода данных используются также Окно ввода (TextBox), функции InputBox и MsgBox.

Окно ввода используется как для ввода, так и для вывода данных:

- писатель = Text1.Text - переменной "писатель" присваивается значение свойства Text объекта Text1;

- физика = Val(Text1.Text) –
- числовой переменной “физика” присваивается значение свойства Text объекта Text1. Свойство Text объекта TextBox хранит значение в символьном виде, поэтому при присвоении значение свойства Text объекта Text1 переменной оно должно быть преобразовано в переменную числового типа с помощью функции Val;
- Text1.Text = “Лев Толстой как зеркало русской революции” – свойству Text объекта Text1 присваивается текстовая константа;
- Text1.Text = Str(физика) - свойству Text объекта Text1 присваивается числовая переменная. Для перевода числовой переменной из числовой формы в текстовую используется функция Str.

Функция InputBox используется для ввода данных в режиме диалога.

Простейший формат функции:

<имя переменной> = InputBox("<текстовое сообщение>")

писатель = InputBox("Введите фамилию автора романа Война и мир")

физика = Val(InputBox("Введите оценку по предмету"))

Функция MsgBox используется для вывода данных. Простейший формат функции:

MsgBox "<Текстовое сообщение>"

MsgBox "Лев Толстой"

MsgBox "Средний балл успеваемости =" & Str(SB)

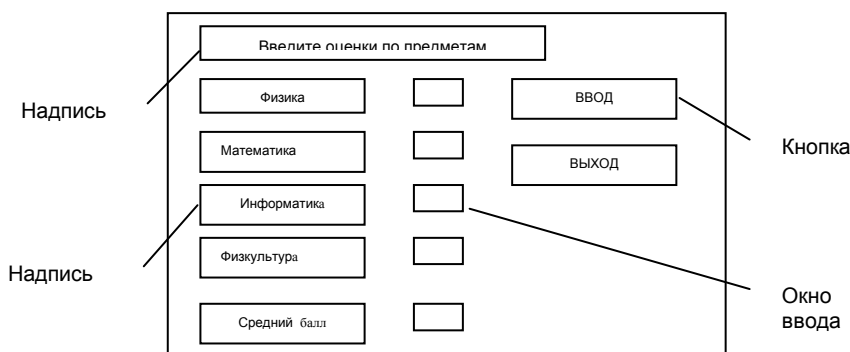
В последнем примере к текстовому сообщению, заключенному в кавычки, прибавляется числовая переменная, которая переводится из числового вида в символьный. В качестве оператора сложения символьных переменных используется символ амперсанд – “&”.

Задание

Создать форму для вычисления среднего балла успеваемости студента. Число предметов обучения - четыре. На форме разместить пять окон для ввода/вывода данных (TextBox), 6 надписей (Label) и две кнопки (CommandButton)

1. Разработайте проект вычисления среднего балла успеваемости студента согласно примеру рис.3.1

2. Разработайте форму для вычисления площади поверхности и объема фигуры по заданию преподавателя.



р. 3. Пример размещения элементов на форме

Порядок выполнения работы

1. Запустите программу Visual Basic 6.
2. Создайте новый проект командой File, New Project.
3. Присвойте форме имя и заголовок (Name, Caption) в окне Свойств (Properties).

4. Разместите на форме надписи (Label) и для каждой из них определите свойства:

- имя (Name);
- текст выводимый в окно метки (Caption);
- выравнивание текста – по центру (Alignment, Center).

5. Разместите на форме окна для ввода оценок, используя элемент управления TextBox. Для каждого окна определите свойства:

- имя (Name);
- начальное значение поля – пусто (Text, удалите информацию);

6. Разместите командные кнопки, используя элемент управления CommandButton. Для каждой кнопки определите свойства:

- имя (Name);
- наименование (Caption);

Остальные свойства объектов принимаются по умолчанию.

7. Откройте окно программы (View Code) и введите текст программы:

8. а) Запишите текст программы в обработчик события Click для кнопки “ВВОД” (имя кнопки cmdVvod):

откройте список объектов и выберите объект cmdVvod. Откройте список свойств и методов и выберите свойство Click. Введите текст программы, например:

```
fiz = Val(txtFizika.Text)
```

Здесь fiz – имя переменной, Val – имя функции преобразования символьной переменной в числовую переменную, txtFizika.Text – значение свойства Text поля ввода txtFizika - оценки по физике .

Аналогично вводятся оценки и по другим предметам обучения. Затем введите формулу для вычисления среднего балла, например,

$Sb = (Fiz + Mat + Inf + Fizk) / 4$

Значение среднего балла присвойте свойству Text окна “Средний балл” (имя окна ввода txtSrBall:

```
txtSrBall.Text = Str(Sb)
```

Функция Str преобразует числовую переменную в символьную.

Пример программы:

```
Private Sub cmdVvod_Click()  
Cls  
fiz = Val(txtFizika.Text)  
mat = Val(txtMatematika.Text)  
inf = Val(txtInformatika.Text)  
fizk = Val(txtFizkultura.Text)  
Sb = (fiz + mat + inf + Fizk) / 4  
txtSrBall.Text = Str(Sb)  
End Sub
```

б) Напишите текст программы для кнопки “ВЫХОД”, имя кнопки cmdExit:

```
Private Sub cmdExit_Click()  
End  
End Sub
```

9. Сохраните проект на диске R: в отдельной папке, например, ЛабРабота2. Введите команду Save As ..., укажите имя файла, например, Успеваемость и щелкните по кнопке Сохранить.

10. Запустите программу для отладки: Run/Start.

11. После отладки сохраните проект.

12. Теперь можно приступить к художественному оформлению проекта. Для этой цели можно использовать такие свойства объектов, как BackColor – цвет фона, ForeColor – цвет символов в форме или цвет символов объектов TextBox и Label, Font – шрифт.

13. Сохраните оформленный проект на диске.

14. Выведите текст проекта на печать командой File/Print, установите флажки Current Project и Code и нажмите клавишу ОК.

Требования к оформлению отчета:

Отчет должен содержать тему, цель занятия, распечатку программы или текст программы, распечатку формы или ее рисунок, описание свойств объектов, установленных на форму, ответы на контрольные вопросы.

Задания

1. Разработать проект, запрашивающий дату рождения пользователя и рассчитывающий, в какой день им было прожито 5000 дней.

2. Разработать проект, запрашивающий у пользователя год его рождения и год его мамы, выдающий результат о возрасте мамы в момент рождения пользователя.

3. Разработать проект, запрашивающий у пользователя имя в TextBox и в том же TextBox приветствующий его.

4. Разработать и оформить проект, запрашивающий и выводящий в одном окне длину одного катета прямоугольного треугольника, в другом окне - длину второго катета, в третьем - длину гипотенузы.

5. Разработать проект, запрашивающий три стороны треугольника a, b, c и вычисляющий его площадь по формуле Герона.

6. Разработать проект, в двух текстовых окнах которого запрашиваются два разных целых числа, а по нажатию командной кнопки они меняются местами.

7. Разработать проект, запрашивающий у пользователя валютный курс на сегодняшний день.

8. Разработать проект, запрашивающий у пользователя длину ребра куба.

9. Разработать проект, определяющий количество цифр во введенном целом сначала четырехзначном, а затем пятизначном. Положительном числе и выводящий последовательно эти самые цифры с комментариями.

10. Создать и оформить проект, выводящий какой-нибудь из неправильных английских глаголов.

Контрольные вопросы

1. Перечислите основные свойства формы.
2. Перечислите основные свойства метки (Label).
3. Перечислите основные свойства окна ввода/вывода (TextBox).
4. Перечислите основные свойства и события кнопки (CommandButton).
5. Приведите пример присвоения значения свойства объекта переменной.
6. Приведите пример присвоения значения переменной свойству объекта.
7. Приведите пример использования функции InputBox.
8. Приведите пример использования функции MsgBox.

Лабораторная работа №4. Разработка меню пользователя.

Цель работы: Закрепить знания в разработке простых форм. Приобрести навыки в разработке меню пользователя, создании родительских (MDI-форм) и дочерних форм.

Краткие теоретические сведения

Разработка меню позволяет сделать приложение с более дружелюбным интерфейсом. VB позволяет иметь до шести уровней вложенности меню. Большое количество уровней тоже не совсем удобно. Рекомендуется использовать не более 3-х уровней вложенности. VB позволяет создавать иерархическое меню. Первый уровень – горизонтальное меню (строка). Второй и последующие уровни – всплывающие меню.

Visual Basic 6.0 имеет удобное средство для разработки меню – Menu Editor, которое вызывается командой Tools / Menu Editor или комбинацией клавиш CTRL – E.

Диалоговое окно Menu Editor приведено на рис. 4.1.

Строка ввода Caption служит для ввода наименования пункта меню, выводимого на экран. После нажатия клавиши ОК или щелчка мыши введенное наименование появляется в окне редактора. Если перед одним из символов наименования пункта меню поставить символ “&” - амперсанд, то появится возможность вызывать пункт меню по нажатию данной клавиши (горячей клавиши). Символ, перед которым стоит знак “&”, подчеркивается.

Строка ввода Name служит для ввода имени пункта меню, которое будет использоваться в программе для обработки событий. Перед именем пункта меню рекомендуется ставить префикс mnu, например, mnuFile.

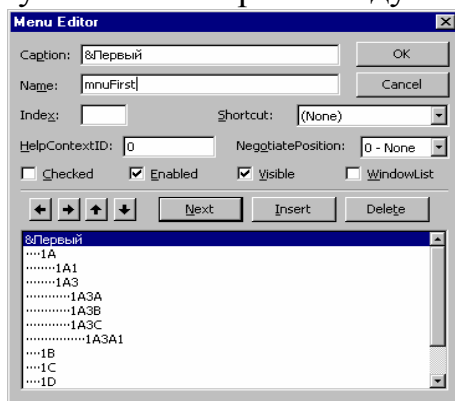


Рис. 4.1. Редактор меню Menu Editor

Программа не позволит пользователю выйти из редактора, пока всем пунктам меню не будут присвоены имена.

Окно Index используется в том случае, если имеется несколько пунктов меню с одинаковыми именами или надо сделать пункты меню частью массива элементов управления.

Окно Shortcut позволяет назначить каждому пункту меню комбинацию клавиш для быстрого вызова команд меню: Ctrl + клавиша, Shift + клавиша и др. При открытии списка появится список быстрых клавиш, из которого надо выбрать нужный.

Окно HelpContexID обеспечивает ввод идентификатора, который используется в электронной справочной системе для выдачи контекстно-зависимой справки по вашему приложению.

Окно NegotiatePosition – служит для определения способа отображения меню на экране, когда один из связанных объектов приложения активен: не показывать, слева, справа, по центру.

Флажок Checked. Если значение данного свойства равно True, то возле соответствующего пункта меню появляется галочка. Это сигнализирует о

том, что соответствующий параметр выбран.

Флажок `Enabled`. Определяет доступность данного пункта меню. Если его значение `False`, то пункт меню недоступен.

Флажок `Visible`. Данное свойство определяет, будет ли виден на экране соответствующий пункт меню. При разработке приложения можно предусмотреть несколько наборов меню, которые должны появляться на экране в соответствующие моменты времени. Например, если в приложении не открыто ни одно окно, меню `Window` не должно появляться на экране.



Флажок `WindowList`. Данный флажок служит для автоматического формирования списка открытых окон, являющихся элементами многодокументного интерфейса (MDI). Установка этого флажка для элемента управления меню верхнего уровня приведет к тому, что в данном элементе будет автоматически формироваться динамический список всех активных дочерних окон.



Кнопка `Next` предназначена для добавления новых пунктов меню.

Кнопка `Insert` позволяет вставить поле для ввода нового пункта меню.

Кнопка `Delete` служит для удаления выделенного пункта меню.

Пункты меню, близкие по назначению целесообразно группировать, отделяя их от других пунктов меню горизонтальной чертой – разделительная линия (`Separator bar`). Эта черта создается так же, как и другие элементы управления меню, но вместо наименования пункта меню (свойство `Caption`) вводится дефис (-). Имя данному пункту меню можно присвоить произвольно.

Изменение уровня вложенности элементов меню осуществляется с помощью кнопок  и . Первая кнопка понижает уровень, вторая – повышает.

Кнопки  и  служат для перемещения выделенного пункта меню. Уровень вложенности элемента управления при этом не изменяется.

Понятие о MDI-форме

MDI-форма - это многодокументный интерфейс, предназначенный для организации взаимодействия нескольких независимых форм.

MDI-форма является родительской формой или контейнером для других (дочерних) форм. Создание MDI-формы осуществляется командой `Project\Add\ MDI Form`.

При запуске программы с MDI-формой программа автоматически устанавливает размеры дочерних окон, которые могут оказаться меньше, чем при настройке, и поэтому часть объектов активной формы может быть невидимой. Чтобы избавиться от этого недостатка, необходимо в обработчике события `Load` каждой формы явно указать размеры и положение формы в окне, например:

```
Private Sub Form_Load()  
Me.Height = 2745
```

```

Me.Width = 3090
Me.Top = (MDIForm1.ScaleHeight - Me.Height) / 2
Me.Left = (MDIForm1.ScaleWidth - Me.Width) / 2
End Sub

```

Если одна из форм максимизируется, то и все последующие открываемые форму будут развернуты на все окно. Чтобы этого не происходило, необходимо при закрытии максимизированной формы приводить ее размеры в нормальное состояние. Для этого в обработчик события кнопки Выход надо поместить следующий код:

```

Private Sub mnuExit_Click()
Me.WindowState = 0
Unload Me
End Sub

```

Работа с дочерними формами

Чтобы обычная форма стала подчиненной MDI-форме (дочерней формой), значение ее свойства MDIChild необходимо установить в True.

Дочерние формы показываются с помощью метода Show, например:

```

Private Sub mnuVvod_Click()
frmVvod.Show
End Sub

```

Управление открытыми формами

установите флажок WindowList для элемента меню Окно;

введите меню второго уровня в меню Окно: каскадом, горизонтально, вертикально. Управление окнами осуществляется с помощью метода Arrange. Кроме того, VB имеет четыре константы для управления окнами: VbCascade – размещение окон каскадом, VbTileHorizontal – размещение окон горизонтально, VbTileVertical – размещение окон вертикально, VbArrangeIcons – пиктограммы всех минимизированных окон располагаются по нижнему краю родительской формы. Последнее свойство реализуется с помощью кнопки системного меню Минимизация.

поместить в обработчики события Click этих пунктов меню объектный код следующего вида:

имяMDI-формы.Arrange константаVisualBasic

Например:

```

Sub mnuCascade_Click
MDIForm1.Arrange VbCascade
End Sub
Sub mnuHorizontal_Click
MDIForm1.Arrange VbTileHorizontal
End Sub
Sub mnuVertical_Click
MDIForm1.Arrange VbTileVertical
End Sub

```

Чтобы указать форму, которая должна стартовать первой, щелкните правой кнопкой мыши в окне проекта и выберите из контекстного меню команду Project Properties. На вкладке General в окне Startup Object выбрать

Задание

1. Разработайте форму для вычисления параметров геометрической фигуры отличной от геометрической фигуры, использованной в лабораторной работе № 3.

2. Добавьте в проект MDI-форму и разработайте меню пользователя для управления формами.

Дополнительное задание

Разработайте контекстное меню для управления размером и начертанием шрифта в форме.

Указания к выполнению задания

1. Загрузите программу лабораторной работы №3.
2. Добавьте новую форму командой Project, Add Form.
3. Разработайте в новой форме пользовательский интерфейс для вычисления параметров другой геометрической фигуры (площади поверхности и объема).

4. Добавьте в проект MDI-форму командой Project, Add MDI Form.

5. Присвойте свойству MDIChild дочерних форм значение True.

6. Разработайте в MDI форме меню пользователя.

Возможная структура меню:

Файл

Создать

Открыть

Сохранить

Печать

Выход

Вычисление

Параллелепипед

Площадь

Объем

Цилиндр

Площадь

Объем

Шар

Площадь

Объем

Помощь

О программе

Автор

Информация

7. Добавьте в главное меню пункт “Окно”, настройте его на автоматическое формирование списка открытых окон, напишите программу для управления окнами;

8. Распечатайте текст программы родительской и дочерней форм, разработанных на текущем занятии.

9. Оформите отчет.

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, задание, тексты программ, структуру меню и ответы на контрольные вопросы.

Задания

1. Усовершенствовать текстовый редактор панелью инструментов для работы с файлом – Открыть, Сохранить, Выход.

2. Дополнить текстовый редактор процедурами поиска и замены фрагментов текста.

3. Создайте проект «Зоопарк».

4. Разработайте возможность вывода на печать теста и результатов пользователя.

5. Создайте проект «Пожиратели звезд»

6. Дополните проект из задания 3 строкой состояния.

7. Дополните проект «Пожиратели звезд» строкой выполнения процесса.

Контрольные вопросы

1. Что такое родительская форма?

2. Что такое дочерняя форма?

3. Как добавить в проект простую форму, MDI форму?

4. Как установить клавишу ускоренного вызова?

5. Как установить стартовую форму?

6. Как разработать меню пользователя?

Лабораторная работа №5. Разработка циклических программ.

Цель работы. Закрепить навыки в разработке простых приложений, а также разработке циклических программ с использованием операторов IF, For/Next, Select Case, While/Wend. Приобрести навыки в использовании функций пользователя.

Краткие теоретические сведения

Вычисление определенного интеграла

Сущность метода численного интегрирования состоит в том, что отрезок интегрирования $[a; b]$ разбивают на n элементарных отрезков $[x_{i-1}, x_i]$ и заменяют интеграл функции $f(x)$ суммой прямоугольников $S_i = f(t_i) \cdot x_i$

(рис. 5.1), где $f(t_i)$ — значение функции в некоторой точке (t_i) внутри i -го отрезка; Δx_i — длина отрезка, $\Delta x_i = x_i - x_{i-1}$, то есть

$$S_n = S_1 + S_2 + \dots + S_{n-1} + S_n + R = \sum_{i=1}^n f(t_i) \Delta x_i + R \quad (5.1)$$

Здесь S_n - интегральная сумма, R - погрешность усечения. Предел этой суммы при неограниченном увеличении числа отрезков разбиения, когда длина наибольшего из элементарных отрезков Δx_i стремится к нулю, есть интеграл от функции $f(x)$ на отрезке $[a; b]$:

$$\int_a^b f(x) dx = \lim_{\max \Delta x_i \rightarrow 0} \sum_{i=1}^n f(t_i) \Delta x_i$$

Простейший метод численного интегрирования — метод средних прямоугольников. В нем используется замена интеграла интегральной функцией (5.1). В этом случае все отрезки Δx_i равны между собой, а в качестве $f(t_i)$ принимается значение функции в середине отрезка:

$$\int_a^b f(x) dx = \sum_{i=1}^n h f(x_{i-1/2}) = \sum_{i=1}^n h f(a + hi - h/2) = \sum_{i=1}^n h_i f(a + (2i-1)h/2),$$

где h — длина элементарного отрезка, $h = (b-a) / n$; $x_{i-1/2}$ — значение аргумента в середине i -го отрезка, $x_{i-1/2} = a + (2i - 1)(b - a)/(2n)$.

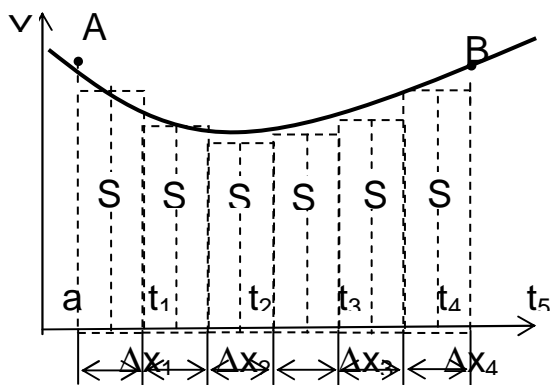


Рис. 5.1. Вычисление определенного интеграла методом средних прямоугольников

Если значение функции вычисляется при x равном значению аргумента на границах отрезков деления то получают, соответственно, метод левых и метод правых прямоугольников.

В других методах вычисляют не площадь прямоугольника, а площадь криволинейной трапеции, заменяя функцию $f(x)$ на элементарном отрезке интерполирующей функцией.

В зависимости от используемой интерполирующей функции различают методы трапеций, Ньютона, Симпсона, Ромберга.

Если в качестве интерполирующей функции используется линейная функция, то получаем метод трапеций:

$$\int_a^b f(x) dx = \frac{1}{2} \sum_{i=1}^n h_i (f(x_i) + f(x_{i-1})) + R \quad (5.2)$$

Формула трапеций (5.2) после преобразования принимает вид:

$$S = h \left(\frac{y_0 + y_n}{2} \right) + \sum_{i=1}^{n-1} y_i, \quad (5.3)$$

В методе Симпсона в качестве интерполирующей функции используется многочлен второй степени $ax^2 + bx + c$. Эти методы обеспечивают получение более точных результатов при меньшем числе

шагов. Формула Симпсона записывается следующим образом:

$$S = \frac{h}{3}(y_0 + y_{2m} + 2(y_2 + y_4 + \dots + y_{2m-2}) + 4(y_1 + y_3 + \dots + y_{2m-1})) \quad (5.4)$$

Для достижения заданной точности применяют метод двойного прохода (пересчета). По двум смежным результатам вычисляется ошибка интегрирования. Для оценки ошибки интегрирования применяют приближенные формулы (принцип Рунге):

$$\Delta = |I_n - I_{2n}|/3 \text{ – для формулы трапеций;} \quad (5.5)$$

$$\Delta = |I_n - I_{2n}|/15 \text{ – для формулы Симпсона.} \quad (5.6)$$

Здесь I_n – значение интеграла при разбиении интервала интегрирования на n частей, а I_{2n} – значение интеграла при разбиении интервала интегрирования на $2n$ частей.

Для оценки ошибки интегрирования в методах левых, средних и правых прямоугольников можно также воспользоваться формулой (5.5).

Вычисление корней уравнения

Вычисление корней уравнения численными методами осуществляется в два этапа: отделение корней и уточнение значений корней на отрезке отделения.

На первом этапе проводится отделение корней уравнения, то есть определение отрезков, на которых находится один и только один корень. Для этой цели проводится табулирование функции с достаточно большим шагом и определяются отрезки, на которых функция меняет знак. Смена знака функции контролируется с помощью выражения $f(x) \cdot f(x+dx) < 0$. Значения границ отрезков отделения следует записать в двумерный массив.

На втором этапе проводится уточнение значения функции на отрезках отделения методом деления отрезка пополам. Подпрограмму уточнения значения функции выполнить в виде процедуры. В этом случае можно совместить процесс отделения корней с процессом уточнения значения корня.

Функция Format

Данная функция предназначена для управления выводом информации на экран или печатающее устройство с заданным числом десятичных знаков, а также с использованием различных разделителей.

Синтаксис функции Format:

Print Format (значение, “шаблон”)

Например: Print Format (1124.75786, “#####.##”)

Символ # резервирует место под цифру, точка отделяет целую часть от дробной. В данном примере зарезервировано шесть знаков в целой части числа и два знака - в дробной части числа. Особенность использования такого пользовательского шаблона состоит в том, что число не должно быть равно нулю. Если число равно нулю, то возникает ошибка

периода выполнения и выполнение программы прерывается.

Задание

1. Разработайте форму для вычисления определенного интеграла с заданной точностью тремя способами, выбранными из известных способов: методом левых прямоугольников, методом правых прямоугольников, методом средних прямоугольников, методом трапеций, методом Симпсона. Сравните полученные результаты по числу циклов, необходимых для достижения заданной точности.

2. Разработайте форму для отделения и уточнения корней алгебраических и трансцендентных уравнений.

Указания к выполнению задания

1. Выполняется одно из заданий по указанию преподавателя.
2. Исходные данные вводить с помощью элемента управления MsgBox.
3. При разработке программ подынтегральная функция и заданная функция для отделения и уточнения значений корней на отрезке отделения задаются в виде функций пользователя.
4. Программа должна позволять вычислять две – три функции. Выбор функции осуществлять с помощью меню. Вычисляемую формулу выводить на экран. Результат вычислений и результат сравнения методов вычисления интеграла выводить в объект TextBox.
5. Поиск корней уравнения осуществлять в два этапа: сначала отыскивается первый отрезок отделения и уточняется значение корня на этом отрезке методом деления отрезка пополам, затем продолжается поиск очередного отрезка отделения. Границы отрезка отделения и значение корня выводить на экран в окно MsgBox. Продолжение поиска должно осуществляться после закрытия окна MsgBox.

Варианты заданий:

а) для вычисления определенного интеграла:

№ вар	1	2	3	4	5	6	7	8	9	10
Задания	1,2, 3,	1,2, 4	1,2, 5	1,3, 4	1,3, 5	1,4,5	2,3, 4	2,3, 5	1,2, 4	1,2, 4

Здесь 1– метод левых прямоугольников, 2 – метод правых прямоугольников, 3 – метод средних прямоугольников, 4 – метод трапеций, 5 – метод Симпсона.

б) для отделения и уточнения значения корня на отрезке:

Задания					
1	$2^x-3x-2=0$ $(0,5)^x+1-(x-2)^2=0$	5	$\arctg x +2x-1=0$ $(x+2)\log_2(x)-1=0$	9	$3x+5x-2=0$ $0,5^x+1-(x-2)^2=0$
2	$3^x+2x-3=0$ $x^2-4+0,5^x=0$	6	$2e^x-2x-3=0$ $\cos(x+0,5)-x^3=0$	10	$(x-3)\cos x-1=0$ $x^4-x^3-2x^2+3x-3=0$
3	$2x^3-9x^2-60x+1=0$ $5\sin x-x+0,5=0$	7	$\text{arcctg}(x-1)+2x-3=0$ $(x-1)^2x-1=0$	11	$3x^4-8x^3-18x^2+2=0$ $(x-2)^2\lg(x+11)-1=0$
4	$e^{2x}-2x+1=0$ $x^2\cos 2x+1=0$	8	$3^x-2x-5=0$ $x\lg(x+1)-1=0$	12	$3^x+2+x=0$ $(x-4)^2\log_{0,5}(x-3)+1=0$

13. Найти значение выражения $1*1+2*2+\dots+n*n$.
14. Составить из двух таблиц 3-ю упорядоченную по возрасту.
15. Найти максимальное число из трёх.
16. Найти максимальное число из четырёх.
17. Количество букв "а" в тексте.
18. Вычислить среднее арифметическое таблицы.
19. Степень числа.
20. Вычислить факториал числа.

Требования к оформлению отчета:

Отчет должен содержать тему, цель занятия, задание, рисунок формы, текст программы и схему алгоритма.

Контрольные вопросы:

1. Поясните принцип работы программы по схеме алгоритма.
2. В чем заключается метод двойного прохода при вычислении определенного интеграла с заданной точностью?
3. Как выполнить отделение корней уравнения численным методом?
4. Поясните алгоритм уточнения корня методом деления отрезка пополам.

Лабораторная работа №6. Использование массивов элементов управления и сетки MSFlexGrid

Цель работы: изучить способы создания массивов элементов управления и их использования для вывода информации. Изучить основные свойства элемента управления MSFlexGrid (сетки) и способы использования ее для вывода информации.

Краткие теоретические сведения

Массив элементов управления представляет собой группу элементов управления одного типа, имеющих одно и то же имя и отличающихся индексом. Обычно его создают на этапе разработки программы. В большинстве случаев для этого достаточно создать массив из одного элемента следующим образом:

- 1) поместить элемент управления в форму;
- 2) задать свойство Name;
- 3) свойству Index присвоить значение 0.

Для добавления элемента управления во время выполнения программы используются методы Load и Add. Метод Load позволяет добавлять элементы управления к существующему элементу управления, метод Add позволяет не только добавлять новые элементы управления, но и создавать новые элементы. Метод Add на занятии не исследуется. Синтаксис метода Load: Load <имя_элемента_управления(индекс)>, где индекс – номер загружаемого элемента. Для вновь созданного элемента управления свойство Visible равно False, свойства Top и Left имеют такие же значения, что и у родительского элемента управления.

Например, для создания 10 элементов управления Label2 может использоваться следующую программу:

```
For i=1 to 10
Load Label2(i)
Label2(i).Visible=True
Label2(i).Top=Label2(i-1).Top+Label2(0).Height
Next i
```

Доступ к конкретному элементу управления в массиве производится по его имени и индексу, например, Label2(5).Caption="Привет!".

Элемент управления MSFlexGrid предназначен для отображения и обработки табличных данных. Он позволяет сортировать, объединять и форматировать таблицы строковых данных. Для добавления сетки MSFlexGrid на панель элементов управления, необходимо в меню Project/Components установить флажок у элемента "Microsoft FlexGrid" и нажать ОК.

Основные свойства объекта MSFlexGrid: Cols/Rows – количество столбцов/строк в сетке; FixedCols/FixedRows – количество фиксированных столбцов/строк в сетке; TextMatrix (строка, столбец) As String – содержимое ячейки на пересечении строки

(Row) и колонки (Col), то же значение имеет свойство Text..

Задания

1. Исследование свойств массива элементов управления:

1.1. Разработать форму для добавления и удаления элементов управления на форму (см. конспект лекций);

1.2. Составить таблицу значений функции согласно варианту. Отрезок табулирования и шаг табуляции выбрать самостоятельно в области определения функции. Результаты записать в массив и вывести на экран с помощью массива элементов управления Label.

2. Исследование свойств элемента управления MSFlexGrid:

2.1. Разработать форму для ввода данных в выделенную ячейку (см. конспект лекций);

2.2. Составить таблицу значений функции на заданном отрезке. Результаты записать в массив и вывести на экран с помощью сетки.

Указания к выполнению задания

1. Разработка формы для добавления и удаления элементов управления на форму:

2. поместите на форму элемент управления txtText1. Свойству Text присвойте значение Текст0;

3. свойству Index присвойте значение 0;

4. установите на форму две кнопки cmdAdd (Добавить) и cmdDel (Удалить);

5. в обработчик события Click кнопки cmdAdd поместите следующий текст программы: Private Sub cmdAdd_Click () Dim i As Single i = Text1().Count
Load Text1(i)
Text1(i).Text = "Текст" & Str\$(i)
Text1(i).Top = Text1(i - 1).Top + Text1(0).Height
Text1(i).Visible = True
End Sub

6. в обработчик события Click кнопки cmdDel поместите следующий текст программы:

```
Private Sub cmdDel_Click()  
Dim i As Single  
i = Text1().Count - 1  
If i > 0 Then  
Unload Text1(i)  
End If  
End Sub
```

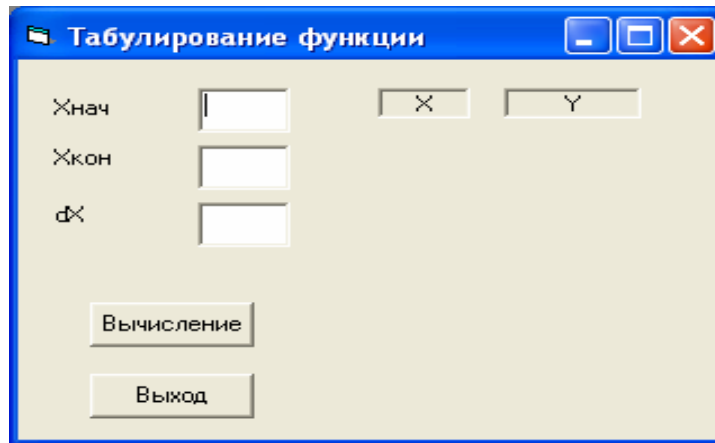


Рис. 6.1. Табулирование функции

2. Табулирование функции

- установить на форму метки и окна ввода для ввода начального (Xнач), конечного (Xкон) значений аргумента и шага табуляции (dX);
- установить две кнопки: cmdCalc (Вычисление) cmdExit (Выход);
- установите две метки lblArgument (X) и lblFunction (Y) для создания массивов элементов управления. Присвойте свойствам Index этих элементов значение 0. Выровняйте текст по центру;

- введите текст программы:

Option Explicit

Dim x As Single, y As Single, dx As Single

Private Sub cmdCalc_Click()

Dim Xnach As Single, Xkon As Single, dx As Single

Dim i As Integer, n As Integer

Xnach = Val(txtText1.Text)

Xkon = Val(txtText2.Text)

dx = Val(txtText3.Text)

n = Int((Xkon - Xnach) / dx) + 1

x = Xnach

For i = 1 To n

Загружаем элементы управления

Load lblArgument(i): Load lblFunction(i)

Вычисляем значение функции

y = Sin(x)

Настраиваем параметры добавленных элементов управления

lblArgument(i).Top = lblArgument(i - 1).Top + lblArgument(0).Height

lblFunction(i).Top = lblFunction(i - 1).Top + lblFunction(0).Height

lblArgument(i).Caption = Str\$(Round(x, 2))

lblFunction(i).Caption = Str\$(Round(y, 2))

lblArgument(i).Visible = True: lblFunction(i).Visible = True

```

x = x + dx
Next i
End Sub

```

Для ввода исходных данных (начала и конца отрезка табулирования, числа шагов или шага табулирования) использовать окна ввода или функцию InputBox.

3. Исследовать свойства элемента управления MSFlexGrid

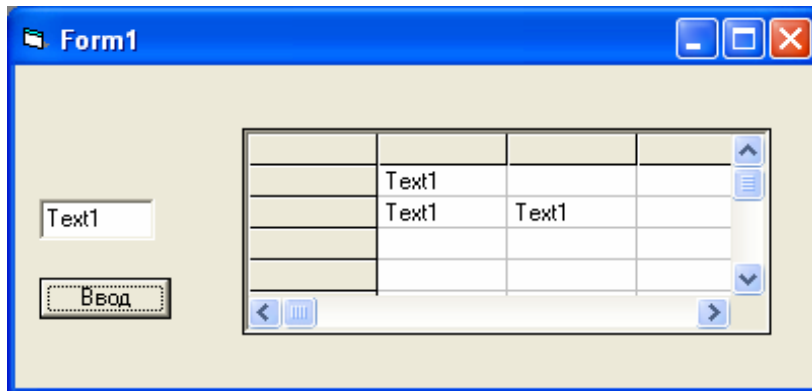


Рис. 6.2. Исследование свойств сетки

работайте

форму согласно рис. 6.2:

- поместите на панель элементов управления сетку: введите команду Project/Components, установите флажок у элемента Microsoft FlexGrid Control 6.0;
- установите сетку на форму. Присвойте ей имя Grid1;
- установите в окне свойств сетки значения требуемых свойств: Rows = 10, Cols = 10, FixedRows =1, FixedCols=1;
- установите окно ввода txtText1 и кнопку cmdVvod;
- запишите в обработчики событий кнопки и сетки следующий текст программы:

```

Private Sub Command1_Click()
MSFlexGrid1.TextMatrix(MSFlexGrid1.RowSel,MSFlexGrid1.ColSel)= _
Text1.Text
End Sub

```

```

Private Sub MSFlexGrid1_LeaveCell() обработчик события очистки
ячейки, выделенной ячейке присваивается белый цвет
MSFlexGrid1.CellBackColor = QBColor(15)
End Sub

```

```

Private Sub MSFlexGrid1_LostFocus()
' очистка ячейки при потере фокуса
MSFlexGrid1.CellBackColor = QBColor(15)
End Sub

```



```
Private Sub MSFlexGrid1_SelChange()
```

```
    'выделенная ячейка закрашивается желтым цветом
```

```
    MSFlexGrid1.CellBackColor = QBColor(14)
```

```
End Sub
```

4. Табулирование функции с использованием сетки

Данный пункт задания выполняется так же, как и п. 2, но вывод данных осуществляется в сетку.

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, задание, рисунки форм и тексты программ, ответы на контрольные вопросы.

Контрольные вопросы:

1. Как создать массив элементов управления на этапе разработки программы?
2. Как добавить элементы управления в процессе работы программы?
3. Как удалить элементы управления с формы?
4. В чем состоят преимущества в использовании массивов элементов управления по сравнению с набором отдельных элементов управления?
5. Каково назначение элемента управления MSFlexGrid?
6. Как добавить новый компонент на панель элементов управления?
7. Перечислите основные свойства сетки.
8. Приведите синтаксис команд ввода данных в сетку и вывода данных из сетки в переменные.

Задания

1. Дана таблица $a[1..n]$. Ввести таблицу $b[1..n]$ отбросив из a каждый второй элемент.
2. Дана таблица $a[1..n]$ из целых чисел. Поставить сначала четные, а потом нечетные элементы.
3. Найти наибольшее кол-во одинаковых элементов.
4. Найти кол-во отрицательных элементов таблицы.
5. Найти максимальный элемент таблицы $a[1..10]$.
6. Получить элементы таблицы, которые находятся между \max и \min .
7. Найти сумму элементов прямоугольной таблицы размером $[n:m]$
8. Найти максимальный элемент прямоугольной таблицы размером $[n:m]$.
9. Дано число n . Выбросить из него все единицы и пятёрки, оставив порядок цифр

ПРИМЕР: 527012 преобразуется в 2702

10. Дано натуральное число n . Выбросить из записи числа все чётные цифры.

11. Перевести число из десятичной в двоичную систему счисления.
12. Перевести число из двоичной в десятичную систему счисления.
13. Дана таблица $a[m,n]$ содержащая числа 0,1,5 или 11. Посчитать кол-во четвёрок
14. Дан одномерный массив. Упорядочить массив, удалив нули со сдвигом влево ненулевых элементов.
15. Уплотнить линейный массив, удалив нули и сдвинув влево остальные элементы.
16. Найти в одномерном числовом массиве элемент, который наибольшее кол-во раз повторяется в массиве. Вывести это число и кол-во его вхождений в массив.
17. Дан одномерный массив размерностью N из положительных и отрицательных чисел. Упорядочить его так, чтобы в начале располагались все отрицательные, а затем все положительные элементы, сохранив порядок следования и не создавая новый массив.
18. Дан прямоугольный целочисленный массив размером $N*N$. Определить, является ли данный массив магическим квадратом, т.е. сумма элементов в строках, столбцах и на главных диагоналях равна.

Лабораторная работа № 7. Построение графиков функций и диаграмм.

Цель занятия: Приобрести практические навыки в построении графиков функции одной переменной, столбиковой диаграммы, круговой диаграммы.

Краткие теоретические сведения

В графическом режиме экран видеомонитора представляет собой набор точек, расположенных по строкам. Каждая точка на экране называется пикселем, число точек по горизонтали и вертикали определяет разрешающую способность экрана. Для работы в среде Windows разрешающая способность должна быть не менее 800x600 пикселей.

Для работы с графикой Visual Basic 6.0 имеет графические объекты, графические элементы управления и графические методы.

К графическим объектам относятся форма (Form) и графическое окно (Picture Box). К этим объектам могут быть применены графические методы.

Графические элементы управления – позволяют помещать на графические объекты линии и геометрические фигуры. К ним относятся элементы управления Line и Shape.

Особо следует выделить элемент управления Image. Он не является ни графическим объектом, ни графическим элементом управления, так как не позволяет применять графические методы, но может использоваться для вставки рисунков.

Графический метод – это метод, который позволяет изображать на объекте данного класса какой-нибудь геометрический элемент, например, точку, линию, окружность и т.д. Графический метод ориентирован на абсолютную или относительную систему координат экрана дисплея.

Абсолютная система координат ориентирована на верхний левый угол экрана со значениями $x=0$; $y=0$, то есть представляет собой IV квадрант прямоугольной декартовой системы координат.

Основной единицей измерения в VB является твип. Твип = 1/1440 логического дюйма. Логический дюйм – это расстояние на форме, которое при печати на принтере будет равно 1 дюйму (1 дюйм = 2,5 см). Используя свойство ScaleMode, можно перейти к другим единицам измерения, например:

1 - Твип (по умолчанию);

3 - Пиксель (пиксель – одна точка на экране монитора, число пикселей определяется установленным разрешением экрана Windows);

Form1.ScaleMode = 3 – установлена единица измерения пиксель.

Для установки другого масштаба, пользовательского, используется метод Scale. Синтаксис метода:

[имяОбъекта]. Scale (x1,y1) – (x2,y2)

где x1,y1 – координаты верхнего левого угла экрана; x2,y2- координаты правого нижнего угла экрана.

Элемент управления Line позволяет рисовать линии различной толщины и стиля.

Этот элемент обладает 15 свойствами. Основными являются X1, Y1, X2, Y2, BorderStyle, BorderWidth и BorderColor.

X1, Y1 – координаты левого конца линии; X2, Y2 - координаты правого конца линии.

BorderStyle - определяет стиль линии:

0 - невидимая; 1 – сплошная; 2 – пунктирная; 3 – пунктирная с коротким штрихом; 4 – штрих пунктирная; 5 – штрих- штрих пунктирная; 6- InsideSolid. Данное свойство работает только при значении свойства BorderWidth=1

BorderWidth - определяет толщину линии и может принимать любые значения кроме нуля.

BorderColor - определяет цвет объекта. Существует четыре способа задания цвета:

- непосредственное задание 16-ричной константой. Например: &H00000000& - черный цвет; &H000080FF& - красный цвет;

- использование RGB – функции: RGB (Red, Green, Blue).

RGB – функция формируется из трех цветов: красного, зеленого и синего. Каждый цвет задается числовой константой от 0 до 255. Например: R=100: G=150: B=75

Line.BorderColor=RGB(R,G,B) ‘темно зеленый цвет

- использование констант Visual Basic. Имеется 8 констант: vbBlack - черный; vbBlue - синий; vbCyan - голубой; vbGreen - зеленый; vbMagenta - сиреневый; vbRed - красный; vbWhite - белый; vbYellow – желтый;

- использование функции QBColor (C), где C - цвета от 0 до 15:

Черный	- 0	Темно – серый	- 8
Темно-синий	- 1	Синий	- 9
Темно-зеленый	- 2	Зеленый	- 10
Темно-голубой	- 3	Голубой	- 11
Темно-красный	- 4	Красный	- 12
Темно-сиреневый	- 5	Сиреневый	- 13
Коричневый	- 6	Желтый	- 14
Светло-серый	- 7	Белый	- 15

Объект Line устанавливается на форму во время разработки программы, как и другие объекты управления. Положение объекта Line на форме можно изменить программным путем.

Элемент управления Shape служит для изображения геометрических фигур: квадратов, прямоугольников, эллипсов, окружностей.

Элемент Shape обладает практически теми же свойствами, что и элемент Line, но имеет и ряд специфических свойств. Основные свойства Top, Left, Height, Width, Shape, BorderStyle, BorderWidth, FillStyle, FillColor.

Top, Left, Height, Width – эти свойства аналогичны свойствам других элементов управления. Они определяют положение объекта на форме и его размеры.

Shape – определяет форму объекта.

FillStyle - обеспечивает автоматическое заполнение фигур, построенных с помощью графических методов.

BorderStyle, BorderWidth – определяют стиль контура и толщину линии соответственно. Эти свойства аналогичны соответствующим свойствам объекта Line.

FillColor – определяет цвет заполнения объекта, аналогичен свойству BorderColor объекта Line.

Чтобы использовать режим управления пикселем необходимо установить свойство ScaleMode =3

Число твипов, приходящихся на один пиксель возвращают функции TwipsPerPixelX и TwipsPerPixelY.

Для управления цветом точки используется метод Pset. Синтаксис метода:

Pset(x,y) [, C].

Метод Pset можно использовать для изображения графиков функций, а также для закраски фигур произвольной формы.

Для определения цвета точки используется метод Point.

Синтаксис метода: Object.Point(x,y)

Задание

Построить график функции согласно варианту задания на произвольном отрезке определения функции.

Дополнительное задание

Создать еще одну форму, на которой поместить столбиковую диаграмму по полученным значениям функции. Шаг табуляции принять достаточно большим.

Указания к выполнению задания

Для решения данной задачи, необходимо протабулировать функцию на заданном отрезке с некоторым шагом и найти максимальное и минимальное значение функции на отрезке табулирования. Затем выполнить масштабирование формы в соответствии с вычисленными параметрами.

Построить график функции, используя метод Pset. Для построения графика функции необходимо повторить операцию табулирования функции, и на каждом шаге строить точку. Чтобы график функции был плотным, шаг табулирования следует выбирать достаточно малым, например, 0.01.



Рис. 7.1. Построение графика функций

Разработайте форму согласно

рис.7.1.

- установите на форму элемент управления PictureBox. Присвойте ему имя picGraph1;
- установите кнопки cmdPlot (Построение) и cmdExit (Выход);
- установите метки для обозначения Xнач, Xкон, Шаг, Ymax, Ymin;
- установите три окна ввода txtXnach, txtXkon, txtDx для ввода начальных значений;

Запишите в обработчик события Click кнопки Построение текст программы:

```
Option Explicit
```

```
Dim x As Single, y As Single, m, xn As Single, ymin As Single
```

```
Dim xk As Single, ymax As Single, dx As Single
```

```
Function FNy(x As Single) ' функция пользователя
```

```
y = 2 * Sqr(Abs(x ^ 3)) * Sin(x)
```

```
FNy = y
```

```
End Function
```

```

Private Sub cmdgraf_Click()
Picture1.Cls
Picture1.ScaleMode = 3
xn = Val(Text1(0).Text):   xk = Val(Text1(1).Text):   dx =
Val(Text1(2).Text)
ymax = FNy(xn): ymin = ymax
For x = xn To xk + dx / 2 Step dx
y = FNy(x)
If y > ymax Then ymax = y
If y < ymin Then ymin = y
Next x
Picture1.Scale (xn, ymax)-(xk, ymin) ' масштабирование формы
Picture1.Line (xn, 0)-(xk, 0)        ' ось X-ов
Picture1.Line (0, ymin)-(0, ymax)   ' ось Y-ов
For x = xn To xk + dx / 2 Step dx
y = FNy(x)
Picture1.PSet (x, y), vbRed
Next x
End Sub
Private Sub cmdExit_Click()
End
End Sub

```

Варианты заданий

1. $\text{Sin}(x)$;
2. e^x ;
3. $\text{Sin}(x+1)$;
4. $\text{Cos}(x)$;
5. $\text{Sin}(2x)$;
6. e^{3x}
7. $\text{Cos}(3x-1)$;
8. $\text{Sin}(3x-1)$;
9. e^{2x} ;
10. $\text{Cos}(2x)$;
11. $\text{Sin}(3x)$;
12. $\text{Sin}(x-0.2)$;
13. e^{x-1} ;
14. $\text{Cos}(x+1)$;
15. $\text{Cos}(x)$
16. $\text{Cos}(2x-1)$;
17. $\text{Sin}(0.5x+1)$;
18. $\text{Sin}(x-0.5)$
19. e^{2x+3} ;
20. $\text{Cos}(2x+1)$;

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, задание, тексты программ, ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое графический объект? Назовите графические объекты.
2. Что такое графический метод? Назовите графические методы.
3. Какие графические элементы управления Вам известны?
4. Какие единицы измерения применяются в графических объектах?
5. Расскажите назначение метода Scale. Приведите синтаксис метода.
6. Что такое объект Screen и каковы его основные свойства?
7. Перечислите основные свойства элемента управления Line.
8. Какие способы используются в VB для задания цвета?
9. Приведите формат RGB – функции.
10. Приведите синтаксис метода PSet.

Лабораторная работа № 8. Графические окна. Анимация

Цель занятия. Приобрести практические навыки в использовании графических окон для вывода графической информации и создания анимационных эффектов.

Краткие теоретические сведения

К графическим объектам относятся элементы управления PictureBox и Image.

Элемент управления PictureBox является элементом-контейнером. В него можно помещать другие объекты, а также рисунки. Основными его свойствами являются: Align, Autosize, Picture.

Align определяет положение PictureBox в форме, Autosize - определяет, будут ли размеры элемента управления автоматически изменяться для отображения рисунков различного размера, Picture - позволяет загружать графические объекты и сохранять их, содержит отображаемый графический объект.

Для загрузки графического объекта на этапе разработки проекта щелкните мышью по кнопке троеточие в поле свойства Picture – появится диалоговое окно Load Picture, которое позволяет осуществить поиск и загрузку нужного файла с диска. Например: C:\Windows\облака.bmp

В процессе работы программы загрузку изображений можно осуществлять с помощью функции LoadPicture:

Объект.Picture = LoadPicture("спецификация_файла")

Выгрузка рисунка в процессе работы осуществляется этой же функцией без указания имени файла:

Объект.Picture = LoadPicture()

Элемент управления Image также создан для отображения рисунков. Но в отличие от PictureBox, он не является элементом контейнером. Элемент управления Image не позволяет ни рисовать, ни группировать объекты.

Основными свойствами элемента Image является свойство Stretch и Picture. Свойство Picture аналогично соответствующему свойству элемента PictureBox. Свойство Stretch позволяет устанавливать соответствие между размерами элемента управления и размером рисунка.

Способы создания анимационных эффектов:

- пересчет координат объекта и использование свойств Top и Left объекта или операторов CurrentX, CurrentY для переопределения координат объекта;
- использование метода Move, например: Command1.Move x, y
- использование буфера обмена;
- прямое присвоение значений свойств одного графического объекта другому.

Режим DrawMode

Для перерисовки графических объектов используется режим DrawMode. Существует 15 возможных установок DrawMode. Во всех случаях VB сравнивает значение цвета пикселя на экране со значением цвета пикселя объекта, который рисуется на экране.

Если DrawMode = 7, то результатом его работы будет оператор Xor. DrawMode = 6 – соответствует оператору Not, а DrawMode = 4 определяет работу VB с оператором Not над значениями цветов переднего плана и использует значения этих цветов для рисования.

Если оператор Xor применяется дважды, то происходит восстановление первоначального цвета. Таким образом, повторное воспроизведение графического объекта при установленном режим DrawMode = 7 для формы или окна позволяет стереть построенное ранее изображение без потери другой информации.

Организация пауз

Для организации пауз можно использовать различные приемы: использование пустых циклов, использование системных часов, использование таймера.

Использование пустых циклов:

```
For i=1 To 1000: DoEvents: Next i
```

Использование системных часов:

```
T=Time() или T=Timere()
```

```
While Time() – T < 2: Wend
```

Использование таймера

Visual Basic позволяет устанавливать до 36 таймеров. Основные свойство таймера: Interval и Enabled. Основное событие – Timer.

Рассмотрим пример. На форму установлен таймер. При щелчке мышью по форме свойству Interval таймера присваивается значение 1000, а свойству Enabled – True и вызывается обработчик события Timer таймера. На экран выводятся значения счетчика TimerTimes. Когда значение счетчика TimerTimes станет равным 5, ему присваивается значение 0. Эта процедура повторяется несколько раз и контролируется счетчиком Timeout. Когда значение счетчика Timeout достигнет заданного значения (в примере оно равно 10), то таймер выключается, так как свойству Enabled присвоено значение False

Пример использования таймера

```
Option Explicit
Dim Msg1 As String
-----
Private Sub Timer1_Timer()
Процедура обработки
события Timer
Static TimerTimes As Integer
Static Timeout As Integer
Cls
TimerTimes = TimerTimes + 1
If TimerTimes > 5 Then
CurrentX = 200: CurrentY = 200
TimerTimes = 0
Timeout = Timeout + 1
If Timeout = 10 Then
Timer1.Enabled = False
'Выключение счетчика
End If
Print Msg1
Exit Sub
Else
CurrentX = 200: CurrentY = 200
Print TimerTimes
Exit Sub
End If
End Sub
-----
Private Sub Form_Click()
Msg1 = "          "
Timer1.Interval=1000
```

```

Timer1.Enabled=True 'Запуск счетчика
Timer1_Timer
End Sub

```

Анимация посредством переноса изображений через буфер обмена

Независимо от способа создания рисунка в VB предусмотрена возможность переноса его в другой элемент управления или другие приложения Windows с помощью буфера обмена. Для этого используются методы `SetData`, `GetData()`, `GetForm`, `Clear`.

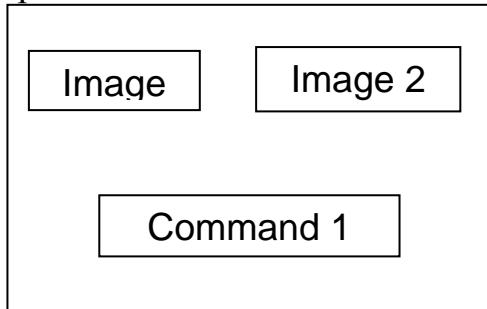


Рис. 8.1. Форма для анимационного перемещения объекта

Метод `SetData` – перемещает данные в объект.

Объект. `SetData [данные],[формат]`

Здесь: Объект - буфер обмена, его идентификатор `Clipboard`;

Данные – указывают, откуда переносятся данные (`ImageBox`, `PictureBox` и др.);

Формат – задает формат исходных данных.

Метод `GetData()` – восстанавливает данные из объекта:
Объект. `GetData [,формат]`

Метод `GetFormat` – возвращает логическое значение, подтверждающее, хранятся ли в объекте данные указанного формата:

Объект. `GetFormat[,формат]`

Пример. Анимационное перемещение изображения, содержащегося в элементе управления `Image1.Picture` в элемент управления `Image2.Picture` (рис.8.1):

```

Private Sub Form_Load()
Image1.Left = 0      ' первоначальная установка объекта Image1
Image1.Top = 0
End Sub
Private Sub Command1_Click()
Dim i As Integer, j As Long
Image2.Left = 1000 ' первоначальная установка объекта Image2
Image2.Top = 1000
Clipboard.Clear ' очистка буфера
Clipboard.SetData Image1.Picture, vbCFBitmap ' загрузка в буфер
' рисунка из объекта Image1
For j = 1 To 1500000: Next j ' пауза
Image2.Picture = Clipboard.GetData() ' загрузка в объект Image2
' рисунка из буфера
For i = 1 To 300      ' движение объекта

```

```

Image2.Left = 1000 + i * 10
Image2.Top = 1000 + i * 10
For j = 1 To 100000: Next j 'пауза
Next I
Image2.Picture = LoadPicture("W:\3dsmax3\ui\atmosApp_24a.bmp")
восстановление изображения из файла
End Sub

```

Анимация посредством присвоения значения свойства одного графического объекта другому

Другой способ анимации состоит в присваивании значения свойства Picture одного графического элемента управления другому.

Пример: Установите на форму четыре объекта Image. В первый, второй и третий элементы загрузите рисунки, например W:\ICQ2002a\Bitmaps\Chat\Smiley0.bmp Smiley12.bmp, Smiley3.bmp. Свойству Visible четвертого объекта Image присвойте значение False. Текст программы поместите в обработчик события Click формы:

```

Private Sub Form_Click()
Image4.Picture = Image1.Picture
Image1.Picture = Image3.Picture
Image3.Picture = Image2.Picture
Image2.Picture = Image4.Picture
End Sub

```

Задание

1. Установите на форму элемент PictureBox и исследуйте его свойства Align, Autosize, Picture.
2. Установите на форму элемент Image и исследуйте его свойства Stretch и Picture.
3. Создайте объект (метку, окно ввода, командную кнопку, картинку, объект Shape), движущийся по заданной траектории.
4. Создайте анимационный эффект посредством присвоения значения свойства одного графического объекта другому объекту.
5. Создайте анимационный эффект посредством переноса изображений через буфер обмена.

Указания к выполнению задания

1. Исследование свойств объекта PictureBox:
 - установите на форму элемент управления PictureBox;
 - исследуйте влияние свойства Align на положение и размеры элемента управления, изменяя значение данного свойства. Для

возвращения размеров объекта в первоначальное состояние присвойте свойству Align значение None и установите требуемые значения свойств Height и Width. Для удаления рисунка из объекта удалите текст Bitmap из строки свойства Picture. Картинку можно взять из следующих каталогов W:\3dsmax3\Images; Windows:\ICQ2002a\Bitmaps\Chat; S:\PIC; S:\PIC\Microsoft\Clipart;

- исследуйте влияние свойства Autosize на размеры рисунка, помещаемого в объект.

2. Исследование свойств объекта Image:

- загрузите в форму картинку;
- изменяя значение свойства Stretch, установите, каким образом влияет данное свойство на размер объекта и картинки при загрузке рисунка;

3. Движение объекта.

- установите на форму объект, например, Shape. Установите его свойства: форму, цвет, размеры;

- поместите на форму таймер, установите значение его свойства Interval равным, например, 100;

- напишите текст программы в обработчик события Timer таймера:

```
Option Explicit
```

```
Dim dx As Integer, dy As Integer, X As Single, Y As Single
```

```
Private Sub Form_Load()
```

```
dx = 150: dy = 150
```

```
Shape1.Left=(Me.ScaleWidth - Shape1.Width)/2
```

```
Shape1.Top= (Me.ScaleHeight - Shape1.Height)/2
```

```
X= Shape1.Left: Y= Shape1.Top
```

```
End Sub
```

```
Private Sub Timer1_Timer()
```

```
Shape1.Left = X + dx
```

```
Shape1.Top = Y + dy
```

```
If Shape1.Top <= 0 Then dy = dy * -1
```

```
If Shape1.Top >= Me.ScaleHeight - Shape1.Height Then dy = dy * -1
```

```
If Shape1.Left <= 0 Then dx = dx * -1
```

```
If Shape1.Left >= Me.ScaleWidth - Shape1.Width Then dx = dx * -1
```

```
X=X+dx: Y=Y+dy
```

```
End Sub
```

При программировании движения объекта установите на форму элемент управления PictureBox и загрузите в него картинку, используя свойство Picture, кнопку или другой элемент управления. Для управления перемещением используйте свойства Top и Left объекта или метод Move.

По пунктам 4, 5 задания реализуйте примеры, приведенные в теоретической части данной лабораторной работы. Для обмена рисунками между элементами управления используйте объекты Image.

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, описание порядка выполнения пунктов задания, ответы на контрольные вопросы, рисунки или распечатки форм, текста программы.

Примеры

Примеры выполнения заданий см. Л1 (с. 138 – 143)

Варианты заданий

Задания первого уровня сложности: набрать и отладить программы, имеющиеся в конспекте лекций или учебном пособии Л1 (с. 138, 143): броуновское движение, светофор и обмен рисунками с помощью буфера обмена.

Задания второго уровня сложности:

1. Самостоятельно разработать программу движения объекта по траектории сложной формы используя метод Move.
2. Разработать программу обмена изображениями между тремя – четырьмя объектами с использованием буфера обмена или обмена изображениями между объектами Image или Picture Box.
3. Нарисовать в Picture Box прямоугольник.
4. Разработать программу «Стрелочные часы».
5. Разработать программу «Солнышко».
6. Разработать программу увеличение и уменьшение размеров объектов в процессе выполнения программы.
7. Разработать программу движения автомашины.
8. Разработать программу движение по кругу.
9. Разработать программу «Прыгающая кнопка».
10. Разработать программу «Летающий текст».

Контрольные вопросы

1. Поясните назначение элементов управления Picture Box и Image? В чем разница между этими элементами управления?
2. Приведите синтаксис команды обращения к свойствам элементов управления Picture Box и Image.
3. Как загрузить графический объект в элементы Picture Box или Image в процессе выполнения программы?
4. Поясните алгоритм создания анимационных эффектов.

5. Какие свойства и методы используются для организации движения объектов?
6. Как организовать паузу при воспроизведении объекта на экране?
7. Какие методы используются для обмена данными через буфер обмена?
8. Как организовать обмен данными между графическими объектами для создания анимационного эффекта?

Лабораторная работа №9. Дополнительные элементы управления для разработки интерфейса.

Цель. Приобрести навыки в использовании дополнительных элементов управления для разработки интерфейса пользователя.

Задание

1. Разработайте форму для демонстрации графиков элементарных функций. Форма должна позволять выводить на экран график трехфазного тока в каждой фазе отдельно и в любых сочетаниях, а также графики экспоненциальной, гиперболической функций и тангенса.
2. Разработайте форму для редактирования текста в окне TextBox: выравнивание текста по левому краю, по правому краю, по центру, а также выбора высоты шрифта и начертания: полужирный, курсив, выделенный, - с использованием таких элементов управления, как флажки, переключатели и списки.
3. Разработайте форму для демонстрации управления цветом с использованием линеек прокрутки и счетчиков.

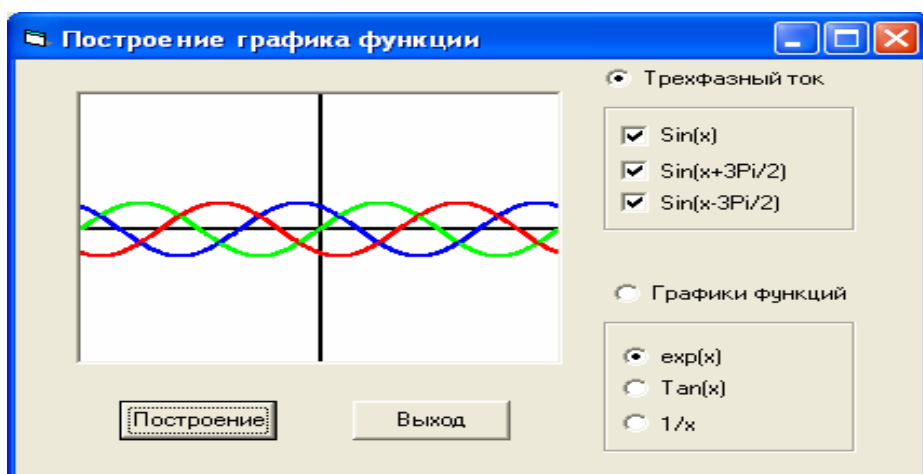


Рис. 9.1. Использование элементов управления Check Box и Option Button

Порядок выполнения работы:

1. При выполнении задания 1 установите на форму элемент управления PictureBox, флажки, переключатели и кнопки согласно рис. 9.1. Выполните масштабирование элемента управления PictureBox симметрично относительно центра элемента управления, например следующим образом:

Picture1.Scale (-2 * pi, 5)-(2 * pi, -5)

Постройте на графике оси координат. Для увеличения толщины линии установите значение свойства DrawWidth элемента управления PictureBox равным двум. Элементы управления CheckBox и OptionButton, заключенные в рамку, можно объявить как элементы массива.

В обработчик события Click кнопки Построение внесите следующий текст программы.

```
Private Sub cmdGraf_Click()
    Picture1.Cls
    Picture1.Scale (-2 * pi, 5)-(2 * pi, -5)
    Picture1.Line (-2 * pi, 0)-(2 * pi, 0)
    Picture1.Line (0, -5)-(0, 5)
    If Option4 Then
        For x = -2 * pi To 2 * pi Step 0.01
            If Check1(0).Value Then y1 = Sin(x): Picture1.PSet (x, y1), vbGreen
            If Check1(1).Value Then y2 = Sin(x + 2 * pi / 3): Picture1.PSet (x, y2),
vbBlue
            If Check1(2).Value Then y3 = Sin(x + 4 * pi / 3): Picture1.PSet (x, y3),
vbRed
        Next x
    Else
        For x = -2 * pi To 2 * pi Step 0.1
            If Option1(0).Value Then y1 = Exp(x): Picture1.PSet (x, y1), vbMagenta
            If Option1(1).Value Then y2 = Tan(x): Picture1.PSet (x, y2), vbCyan
            If Option1(2).Value Then y3 = 1 / x: Picture1.PSet (x, y3), vbGreen
        Next x
    End If
End Sub
```

End Sub

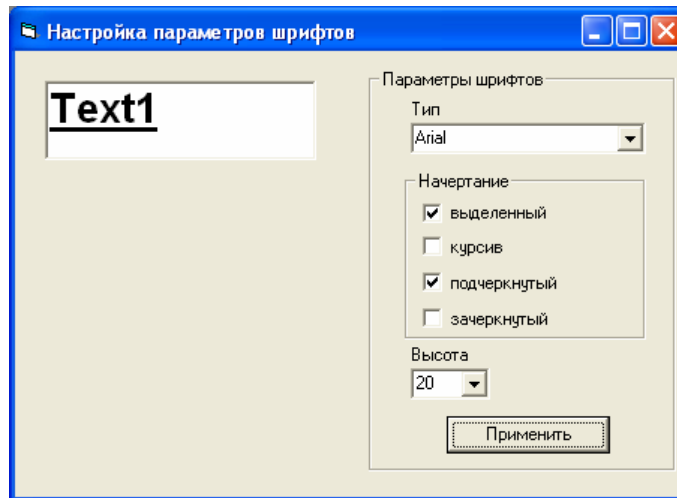


Рис. 9.2. Настройка параметров шрифтов

2. При выполнении задания 2 установите на форму элементы управления согласно рис. 9.2. Заполните списки элементов ComboBox (см. текст программы). Запишите в обработчик события Click кнопки Применить следующий текст программы:

```
Dim Ctl As Control
```

```
-----  
Private Sub cmdStart_Click()  
For Each Ctl In Controls  
If TypeOf Ctl Is TextBox Then  
Ctl.Font.Name = Combo1.Text  
Ctl.Font.Bold = False  
Ctl.Font.Italic = False  
Ctl.Font.Underline = False  
Ctl.Font.Strikethrough = False  
If Check1 Then Ctl.Font.Bold = True  
If Check2 Then Ctl.Font.Italic = True  
If Check3 Then Ctl.Font.Underline = True  
If Check4 Then Ctl.Font.Strikethrough = True  
Ctl.FontSize = Combo2.Text  
End If  
Next Ctl  
End Sub
```

Проверьте работу программы: настройте параметры и щелкните по кнопке Применить – текст в окне Text1 изменит начертание. При вводе нового текста он будет отображаться установленным типом шрифта.

Дополните программу командами выравнивания текста в окне

TextBox.

3. Для выполнения задания по пункту 3 разработайте форму согласно рис. 9.3. Форма предназначена для исследования функции управления цветом RGB. Линейки прокрутки, окна TextBox и счетчики UpDown, предназначены для управления цветом объекта Text1. Элементы управления ScrolBar1, Text2, UpDown1 объявите как массивы элементов управления. Для демонстрации изменения цвета можно использовать любой объект, имеющий свойство BackColor: TextBox, Label, Shape, текущую форму.

При изменении положения ползунка на любой линейке прокрутки синхронно должен изменяться цвет объекта Text1 и одновременно значение кода цвета должно отображаться в окне Text2. Код цвета можно вводить в окно Text2 с клавиатуры или с помощью элемента управления

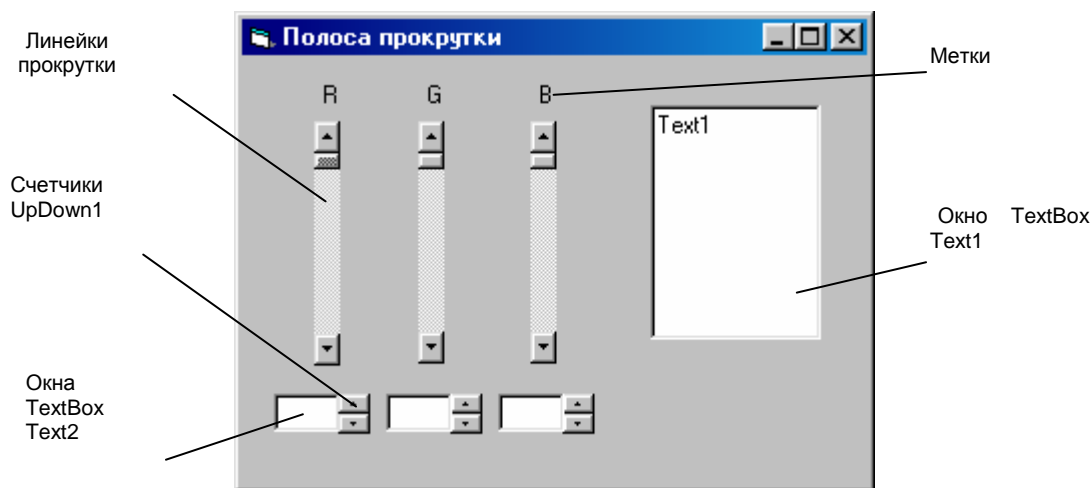


Рис. 9.3. Управление цветом объект

UpDown1. Одновременно с вводом кода цвета в окно Text2 должен меняться цвет объекта Text1 и положение ползунка. Максимальное значение свойств Value элементов управления ScrolBar и UpDown должно быть 255.

При установке элементов управления Text2 и UpDown1 соблюдайте последовательность установки: установите элемент управления Text2(0), установите элемент управления UpDown1(0) и так далее. В этом случае свойства TabIndex элементов управления Text2(i) и UpDown1(i) будут смежными, что облегчит настройку элемента управления UpDown.

Настройка элемента управления UpDown:

- вызовите контекстное меню объекта UpDown1(0) щелчком правой кнопки мыши по объекту;
- щелкните по пункту меню Properties. Открывается окно диалога Property Pages;
- выберите в окне диалога закладку Buddy; установите флажки AutoBuddy и SyncBuddy. При правильной установке свойств TabIndex объектов Text2(0) и UpDown1(0) в окне

Buddy Control появится имя объекта Text2(0), а в окне Buddy Property – слово Default – по умолчанию. Таким образом будут связаны свойства Text элемента управления Text2(0) и свойство Value элемента управления UpDown1(0). Настройте аналогично и другие элементы управления Text2 и UpDown1.

Текст программы запишите в обработчики событий Change элемента управления TextBox и в обработчик события Scrol элемента управления ScrolBox. Текст программы:

Задания

1. По введенному номеру месяца определить его название, время года и загрузить соответствующую картинку.

2. Разработать проект, который по знаку арифметической операции выводит ее название.

3. Разработать проект, который определяет по заданному числу месяца и по дню недели первого числа этого месяца день недели для заданного числа.

4. Разработать программу «Игра в кости»

5. Разработать программу «Угадайка».

Контрольные вопросы

1. Перечислите основные свойства элемента управления CheckBox.

2. Перечислите основные свойства элемента управления Option Button.

3. Перечислите основные свойства элемента управления ScrollBar.

4. Как установить связь между элементом управления ScrollBar и TextBox?

5. Как настроить параметры элемента управления UpDown?

6. Напишите текст программы для управления цветом элемента управления Text1.

7. Перечислите основные свойства раскрывающегося списка (ComboBox).

8. Как внести данные в список на этапе разработки программы и в процессе выполнения программы?

Лабораторная работа №10. Работа с файлами. Использование стандартного диалогового окна Windows.

Цель занятия: Приобрести практические навыки в работе с файлами последовательного доступа.

Краткие теоретические сведения

Текстовые файлы последовательного доступа предназначены для записи и чтения неструктурированных данных. Достоинством файлов последовательного доступа является простота их создания и использования. При необходимости, файл последовательного доступа может быть создан или отредактирован любым текстовым редактором.

Разделителем текста в файлах последовательного доступа является символ возврата каретки, который формируется автоматически при нажатии клавиши Enter.

Для работы с файлами данных используются команды открытия файла, закрытия файла, записи и чтения данных из файла, а также ряд функций, облегчающих работу с файлами.

Для открытия файлов служит команда Open.

Open “спецификация_файла” For { тип файла}[Access{доступ}][Lock{блокировка}] As [#] N [Len=длина]

Опция “Спецификация_файла”, как известно, позволяет указать диск, маршрут, имя и расширение имени файла. Например:

R:\Prognoz\Ucheb\prognoz1.dan

Тип файла указывает на его структуру и способ использования и может принимать следующие значения:

Input – файл последовательного доступа, открыт для чтения;

Output – файл последовательного доступа, открыт для записи;

Append - файл последовательного доступа, открыт для добавления данных;

Binary – двоичный файл открыт для записи и чтения данных;

Random – файл прямого доступа открыт для записи и чтения данных.

Так как пользователь при написании программы в принципе не может знать, сколько каналов занято и каков номер свободного канала, то для определения номера свободного канала следует использовать функцию FreeFile. Функция FreeFile возвращает номер свободного канала.

Для закрытия файлов используется команда Close. Синтаксис команды:

Close [# <номер канала>]

Команда Close с параметром номера канала закрывает указанный канал. Команда Close без параметров закрывает все открытые файлы. С целью надежного сохранения информации рекомендуется использовать вместо команды Close команду Reset. Эта команда, в отличие от команды Close, дает указание операционной системе сбросить содержимое буфера на диск.

Работа с файлами последовательного доступа состоит из двух самостоятельных операций: создания файла и использования файла.

Создание файла последовательного доступа:

Открытие файла ‘ (команда Open или Append с опцией Output)

Запись данных в файл (операторы Write # или Print #).

Закрытие файла ‘ (команда Close)

Чтение данных из файла последовательного доступа:

Открытие файла ‘ (команда Open с опцией Input)

Чтение данных из файла (оператор Input # или Line Input #).

Закрытие файла ‘ (команда Close)

Запись данных в файл последовательного доступа.

Для записи данных в файл последовательного доступа используются операторы Print # и Write #.

При использовании оператора Print числовые данные, записываемые в файл, необходимо преобразовывать в строку символов, особенно это касается вещественных чисел, так как десятичную точку программа воспринимает как разделитель данных. Поэтому при работе с числами предпочтительнее использовать оператор Write #.

Чтение данных из файла последовательного доступа осуществляется операторами Input #, Line Input # .

Оператор Line Input # считывает из файла строку данных. Разделителем данных в файле в этом случае должен быть символ возврата каретки. Строка данных не должна превышать 255 символов.

Оператор Input # имеет следующий синтаксис:

Input # <номер канала>[, “текстовое сообщение”],<список переменных>

Переменные в списке разделяются запятыми.

Пример 10.1. Создание файла последовательного доступа.

```
Open “R:Test.dan” For Output As #1
A$ = “Минск – столица Республики Беларусь”
V%=13875
C!=7.58
Print#1, A$, V%, Str$ (C!)
Close #1
```

Пример 10.2. Использование файла последовательного доступа

```
Open “R:Test.dan” For Input As #1
Input #1, A$, V%, C$
Print A$, V%, Val (C8)
Close #1
```

На форме будет строка следующего вида:

Минск – столица Республики Беларусь 13875 7.58

В данном примере 13875 и 7.58 – числа

Оператор Input # целесообразно использовать в сочетании с оператором Write #.

Задание

Разработать и отладить базу данных “Склад” с использованием файла последовательного доступа (рис.10.1).

Порядок работы

1. Разработайте структуру БД согласно рис. 10.1

Таблица 10.1

N	Наименование	Дата поступления	Номер документа	Количество	Цена	Стоимость

Рис. 10.1. Структура базы данных

2. Разработайте форму согласно рис.10.1

Для хранения базы данных в ОЗУ используйте двухмерный массив $BD(n,5)$. Где n – число записей в базе данных, а 5 – число полей. Номер записи нужен только на экране или на бумаге, в программе хранить его не требуется. Для отображения базы данных на экране воспользуемся сеткой `MSFlexGrid`. Для ввода данных создадим линейку из массива элементов управления.

1. Опишите состав элементов управления на форме (табл. 10.1).
2. Опишите переменные, используемые в программе (табл.10.2)

Напишите текст программы.

Объявление переменных уровня формы:

```
Dim i As Integer, j As Integer, Bd() As String
```

```
Dim n As Integer, Sb As Single, nKanal As Integer
```

Установка начальных параметров при загрузке формы:

```
Private Sub Form_Load()
```

```
Me.Height = 4275
```

```
Me.Width = 8265
```

```
For i = 0 To 5
```

```
Grid1.ColAlignment(i)=3
```

```
Grid1.TextMatrix(0,i) = Label2(i).Caption
```

```
Next i
```

```
Grid1.ColAlignment(6) = 3
```

```
Grid1.TextMatrix(0,5) = "Стоимость"
```

Описание элементов управления

Тип	Имя	Назначение
Label	lblLabel1 lblLabel2()	текст “Число записей” массив элементов управления. Текст – заголовки шапки таблицы ввода данных
TextBox	txtText1 txtText2()	ввод числа записей массив элементов управления. Поля для ввода данных
MSFlexGrid	Grid1	таблица для вывода результатов
Command	cmdVvod cmdSave cmdOpen cmdExit	ввод данных в массив сохранение данных чтение данных с диска (открытие файла) ВЫХОД

Grid1.Row = 0

Grid1.ColWidth(0) = 600: Grid1.ColWidth(1) = 2000

Grid1.ColWidth(2) = 1000: Grid1.ColWidth(3) = 1100

Grid1.ColWidth(4) = 1200: Grid1.ColWidth(5) = 1200

Grid1.ColWidth(6) = 1200

End Sub

Таблица 10.2

Описание переменных

Имя переменной	Тип переменной	Видимость переменной	Комментарий
i, j, k	Integer	Локальная	Используются в качестве переменных цикла
n	Integer	Глобальная	Число записей
Sb	Single	Глобальная	Стоимость
Bd(5,n)	Single	Глобальная	Массив для хранения данных
nKanal	Integer	Глобальная	Номер канала

Текст программы для установки числа строк запишите в обработчик события Change элемента управления txtText1, так как первой операцией при вводе данных необходимо указать число записей.

```
Private Sub txtText1_Change()
```

```
n = Val(txtText1.Text)
```

```
Grid1.Rows = n + 1
```

```
End Sub
```

Программы ввода данных, сохранения и чтения данных с диска запишем в обработчики событий соответствующих кнопок.

Процедура ввода данных.

```
Private Sub cmdVvod_Click()
Dim i As Integer, j As Integer
n = Val(txtText1.Text)
If n = 0 Then
MsgBox "Укажите число записей"
Exit Sub
End If
ReDim Preserve bd(6, n) As String
If n < Val(txtText2(0).Text) Then
n = Val(txtText2(0).Text)
Grid1.Rows = n + 1
ReDim Preserve bd(6, n) As String
End If
i = Val(txtText2(0).Text)
Grid1.Row = i:
For j = 0 To 5
bd(j, i) = txtText2(j).Text
Grid1.Col = j
Grid1.Text = bd(j, i)
Grid1.Visible = True
Next j
st = Val(txtText2(4).Text) * Val(txtText2(5).Text)
Grid1.Col = j
Grid1.Text = Str$(Round(st, 2))
Grid1.Visible = True
bd(6, i) = Str$(st)
For j = 1 To 5
txtText2(j).Text = ""
Next j
If i < n Then txtText2(0).Text = i + 1
End Sub
```

Процедура сохранения данных на диске

```
Private Sub cmdSave_Click()
nKanal = FreeFile
Open "f:\Laborat\VisualBasic\file.dan" For Output As #nKanal
Write #nKanal, n
For i = 1 To n
For j = 0 To 6
Write #nKanal, bd(j, i)
Next j
Next i
Close #nKanal
```

```

End Sub
Процедура чтения данных (открытие файла)
Private Sub cmdOpen_Click()
nKanal = FreeFile
Open "f:\Laborat\VisualBasic\file.dan" For Input As #nKanal
Input #nKanal, n
ReDim bd(6, n)
Grid1.Rows = n + 1
For i = 1 To n
For j = 0 To 6
Input #nKanal, bd(j, i)
Grid1.TextMatrix(i, j) = bd(j, i)
Next j
Next i
Close #nKanal
End Sub
Процедура завершения работы с программой
Private Sub cmdExit_Click()
Unload Me
End Sub

```

Требования к оформлению отчета

Отчет должен содержать тему, цель занятия, задание, тексты программ, ответы на контрольные вопросы.

Задания

1. Создать последовательный файл для канцелярии по учету выпускников.
2. Создать последовательный файл, содержащий имя и адрес каждого сотрудника компьютерной фирмы.
3. Создать файл, в котором содержались бы имена, сведения о сроках уплаты и размере налога для каждого налогоплательщика.
5. Написать программу, которая позволяла бы адвокату вводить в последовательный файл следующие сведения: имя клиента, обвинение, исход дела.
6. Создать файлы хоккейных команд «Черные ястребы» и «Красные крылья», содержащие информацию о каждой из двух команд.
7. Создать последовательный файл академических занятий студентов колледжа.
8. Написать программу, которая создала бы последовательный файл контрагентов фабрики игрушек.
9. Создать последовательный файл, содержащий имя и адрес каждого сотрудника предприятия.

10. Создать последовательный файл для отдела кадров предприятия по учету кадров.

Контрольные вопросы

1. Какие типы файлов данных Вам известны, и чем они отличаются?
2. Приведите синтаксис команды Open.
3. Приведите синтаксис команды Close.
4. Какие команды используются для записи данных в файл последовательного доступа?
5. Какие команды используются для чтения данных из файла последовательного доступа?
6. Какая последовательность команд необходима для создания файла последовательного доступа?
7. Какая последовательность команд необходима для чтения данных из файла последовательного доступа?
8. Расскажите алгоритм разработки базы данных на основе файла последовательного доступа.

Литература:

1. Быков В. Л. Основы программирования на языке Visual Basic 6.0.- Брест, БГТУ, 2002.
2. Г.И. Магданурова. Visual Basic на практике. - Издательская группа BHV, Санкт-Петербург, 2008.
3. А.Ананьев, А. Федоров. Самоучитель Visual Basic 6. - BHV, Санкт Петербург, 2000.
4. С. Браун. Visual Basic 6. - Питер, Санкт Петербург, 2002
5. Брайн Сайлер и Джефф Скоттс. Использование VB 6.0 - М.: СПб.; К.: Издательский дом "Вильямс", 2001.- 832 с.: ил.
6. Волчѐнков Н. Г. Программирование на Visual Basic 6. – М.: ИНФРА, 2000.
7. Справочная система VB.
8. Сафронов И. Visual basic в задачах и примерах. Санкт-Петербург, 2008г

Содержание

Введение.....	3
Лабораторная работа № 1. Работа в среде Visual Basic.....	4
Лабораторная работа № 2. Разработка программ в среде Visual Basic (окно программы).....	11
Лабораторная работа № 3. Создание простой формы.....	15
Лабораторная работа № №4. Разработка меню пользователя.....	19
Лабораторная работа № 5. Разработка циклических программ.....	24
Лабораторная работа № 6. Использование массивов элементов управления и сетки MSFlexGrid.....	29
Лабораторная работа № 7. Построение графиков функций и диаграмм.....	34
Лабораторная работа № 8. Графические окна. Анимация.....	39
Лабораторная работа № 9. Дополнительные элементы управления для разработки интерфейса.....	46
Лабораторная работа № 10. Работа с файлами. Использование стандартного диалогового окна Windows	50
Литература	58
Содержание.....	59

**Бообеков Т.И., Балан Р.К.,
Асанова З.А.**

**Методическое указание
к выполнению лабораторных работ
по дисциплине «Язык программирования Visual Basic 6.0»**

Технический редактор: Ч.А.Жакыпова
Компьютерная верстка: З.А. Чоколокова

Отпечатано в полиграфическом комплексе
ЫГУ им. К.Тыныстанова.
Заказ 331. Тираж 25.
Тел.: (03922) 52696.