КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И.Раззакова

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра **«ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ КОМПЬЮТЕРНЫХ СИСТЕМ»**

СРЕДСТВА ВИЗУАЛЬНОЙ РАЗРАБОТКИ ПРИЛОЖЕНИЙ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ДЛЯ СТУДЕНТОВ СПЕЦИАЛЬНОСТИ 552801.04 «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ»

Бишкек - 2011

«РАССМОТРЕНО»
на заседании кафедры
«Программное обеспечение
компьютерных систем»
Прот.№4 от 04.10.2011г.

«ОДОБРЕНО»
Методическим советом ФИТ
Прот. №3 от 19.10.2011г.

УДК 681.31.01

Составитель **МУСИНА И.Р.**

Средства визуальной разработки приложений. Методические указания к выполнению лабораторных работ / КГТУ им. И. Раззакова; сост. И.Р.Мусина. – Б.: ИЦ «Текник», 2011. – 47 с.

Представлены общие теоретические сведения по средствам визуальной разработки приложений, приведены задания на разработку программ на Visual Basic 2008 в среде .Net Framework, методические указания к их выполнению и варианты заданий для самостоятельной работы.

Предназначено для студентов специальности «Программное обеспечение вычислительной техники и автоматизированных систем» всех форм обучения.

Рисунков – 27, таблиц – 5, библиография – 5 наименований.

Рецензент доцент кафедры ПОКС, к.т.н. Тен И.Г.

Тех. редактор ***Субанбердиева Н.Е.***

Подписано к печати 29.11.2011 г. Формат бумаги 60x84$^{1}/_{16}$.
Бумага офс. Печать офс. Объем 3 п.л. Тираж 30 экз. Заказ 405. Цена 50,5 с.
Бишкек, ул. Сухомлинова, 20. ИЦ “Текник” КГТУ им. И.Раззакова, т.: 54-29-43
e-mail: beknur@mail.ru

Содержание

Введение

На начальном этапе существования компьютерных информационных систем их разработка велась на традиционных языках программирования. Однако по мере возрастания сложности разрабатываемых систем и увеличения запросов пользователей (чему в значительной степени способствовал прогресс в области вычислительной техники, а также появление удобного графического интерфейса пользователя системном программном обеспечении) потребовались новые средства, обеспечивающие значительное сокращение сроков разработки. Это послужило предпосылкой к созданию целого направления в области программного обеспечения – инструментальных средств для быстрой разработки приложений.

Методология разработки информационных систем, основанная на использовании средств быстрой разработки приложений, получила широкое распространение и приобрела название *методологии быстрой разработки приложений – RAD (Rapid Application Development).*

Средства RAD дали возможность реализовать совершенно иную по сравнению с традиционной технологию создания приложений: информационные объекты формируются как некие действующие модели (прототипы), чье функционирование согласовывается с пользователем, а затем разработчик может непосредственно переходить к формированию законченных приложений, не теряя из виду общей картины проектируемой системы. Визуальные инструменты RAD (*средства визуального программирования*) позволяют создавать сложные графические интерфейсы пользователя вообще без написания кода программы. При этом разработчик может на любом этапе наблюдать то, что закладывается в основу принимаемых решений.

Визуальные средства разработки оперируют в первую очередь со стандартными интерфейсными объектами – окнами, списками, текстами, которые легко можно связать с данными из базы данных и отобразить их на экране монитора. Другая группа представляет собой стандартные элементы управления – кнопки, переключатели, флажки, меню и т.п., с помощью которых осуществляется управление отображаемыми данными. Все эти объекты могут быть стандартным образом описаны средствами языка, а сами описания сохранены для дальнейшего повторного использования.

В настоящее время существует довольно много различных визуальных средств разработки приложений. Но все они могут быть разделены на две группы – универсальные и специализированные. Среди универсальных средств одним из наиболее распространенным является Visual Basic. Мы называем средства универсальными потому, что они не ориентированы на разработку только приложений баз данных. С их помощью могут быть разработаны приложения почти любого типа, в том числе и информационные приложения. Причем программы, разработанные с помощью универсальных систем, могут взаимодействовать практически с любыми системами управления баз данных. Это обеспечивается как использованием драйверов ODBC или OLE DB, так и применением специализированных средств (компонентов).

В рамках дисциплины «Средства визуальной разработки приложений» студенты осваивают язык VISUAL BASIC 2008, который позволяет создавать приложения практически для любой области современных компьютерных технологий: бизнес-приложения, игры, мультимедиа, базы данных, интернет-приложения. При этом приложения могут быть как простыми, так и очень сложными, в зависимости от поставленных задач.

В Visual Basic 2008 используются все самые современные методы программирования: объектно-ориентрованная модель, включая наследование визуальных классов, модель составных объектов COM (Component Object Model), технология программных компонентов ActiveX и др. Создание приложений будет осуществляться на платформе .Net Framework.. .NET Framework — это вычислительная платформа, которая упрощает разработку приложений в сильно распределенном окружении Интернета. Платформа .NET Framework предлагает следующие возможности:

- предоставление среды выполнения кода, которая уменьшает конфликты между разными версиями языков, гарантирует безопасное выполнение кода, устраняет проблемы, связанные с переносом сред;
- работу с различными типами приложений, размещенных как в Интернете, так и на локальном компьютере;
- установление стандартов для поддержки платформы .NET Framework другими языками;
- создание непротиворечивой объектно-ориентированной среды программирования, в которой код объекта может храниться и выполняться локально, выполняться локально, но быть распределенным в Интернете или выполняться удаленно.

Данное пособие предназначено для студентов второго курса специальности «Программное обеспечение вычислительной техники и автоматизированных систем», изучающих VISUAL BASIC.NET (дисциплина «Средства визуальной разработки приложений»). В нем представлены задачи по программированию на VB в среде .NET FRAMEWORK, краткие теоретические сведения и указания к разработке программ, приведены задания для самостоятельной работы.

Целью выполнения лабораторных работ является:

- освоение и закрепление навыков программирования на VB.NET;
- освоение навыков визуального программирования;
- развитие навыков построения программ при решении типовых задач, реализующих численные методы (численное интегрирование, решение алгебраических уравнений и систем, поиск корня, экстремума и т.д.);
- освоение технологии работы с текстовыми файлами;
- развитие навыков построения графиков функций;
- освоение элементов объектно-ориентированного программирования.

По окончании изучения дисциплины студент должен уметь создавать приложения на языке VISUAL BASIC в среде VISUAL STUDIO. NET FRAMEWORK, используя визуальные инструменты RAD.

Лабораторная работа №1
Создание простейшей формы

Цель: Научиться создавать формы с простейшими элементами управления.

Задание№1.1

Создание проекта. Работа с одной формой

Постановка задачи. Создать приложение, которое загружается с формы, имеющее название «Привет». В текстовое поле формы должно выводиться сообщение «Привет от ПОВТ». В окне формы вывести 2 рисунка с надписями. Под рисунками расположить 2 кнопки. При нажатии каждой должны меняться надписи под рисунками. Создать на форме третью кнопку для завершения приложения.

Указания к выполнению задания

1. Запустите Visual Basic.NET и выберите команду File=>New=>Project.
Visual Basic.NET отобразит диалоговое окно New Project, в котором вы сможете выбрать тип создаваемой программы.
2. Выберите тип Windows Form Application (Приложение Windows), щелкнув по соответствующему значку.
3. Щелкните в строке Name (Название) и наберите имя нового проекта, а затем щелкните на кнопке ОК.
Visual Basic .NET отобразит пустую форму, именуемую Forml.
4. Поместите курсор мыши над правым нижним углом формы (прямо над маленьким квадратиком, называемым маркером, который будет отображен напротив угла формы) так, чтобы он принял вид двунаправленной стрелки.
Нажмите левую кнопку мыши и перетащите курсор, придав окну формы нужный размер.
5. Выберите команду View=> Tool box, чтобы на своем обычном месте в левой части экрана отобразилась панель Toolbox. (Пропустите этот шаг, если панель Toolbox уже отображена.)
6. В панели Toolbox щелкните на значке Button (Кнопка).
7. Поместите курсор в окно формы, а затем перетащите его, чтобы нарисовать.
Повторите шаги 6 - 7 еще два раза, чтобы нарисовать еще две кнопки.
8. В панели Toolbox щелкните на кнопке Picture Box (Рисунок) и нарисуйте объект в окне формы.
Повторите этот процесс еще один раза, чтобы выделить места для трех рисунков.
9. Щелкните на значке объекта Label (Надпись) и нарисуйте его в окне формы по центру.
Подготовьте аналогичным способом надписи под каждым Рисунком (местом для будущего рисунка).
10. Выделите один из объектов PictureBox.
11. Нажмите клавишу <F4>, чтобы открыть окно Properties.

12. Щелкните на свойстве Image (Изображение), которое относится к категории Appearance (Отображение).
Появится кнопка с тремя точками (**...**).
13. Щелкните на кнопке с тремя точками (**...**).
Откроется диалоговое окно Select Resource.
14. Используя кнопку Import, .выберите нужный вам рисунок из соответствующей папки и щелкните на кнопке Open. Visual Basic .NET его отобразит в области объекта Рисунок.
15. Щелкните на свойстве SizeMode (Размер из категории Behavior), затем на кнопке со стрелкой и в открывшемся списке выберите пункт Stretchlitiage.
Visual Basic .NET увеличит вставленный рисунок до размеров выделенной для него области.
16. Повторите действия 10-14 для второго рисунка.
17. Измените имя кнопки, которая находится по центру, и дайте ей имя новое имя btnExit. Поменяйте свойство text (текст) на кнопке на Exit
В результате на экране вы увидите примерно следующий рисунок:

Рис.1.1

18. Выберите объект Label3 (надпись), щелкнув по нему мышью.
19. Дважды щелкните на свойстве Text (категория Appearance) и удалите установленное по умолчанию значение. Введите нужный вам текст.
20. Для того чтобы при нажатии на кнопку менялась надпись, дважды щелкните по кнопке Button1. В теле открывшейся процедуры между

```
 Private Sub button1_Click(ByVal sender As Object, ByVal
e As System.EventArgs) Handles button1.Click
```
и

End Sub напишите команду:

Label1.Text = "Dance!!!"

Замечание: знак ‘ (одинарная кавычка) используется для комментария в коде.

Эта команда позволяет поменять свойство text объекта **button1**.

21. Проделайте аналогичное (пункты 18 – 20) для изменения надписи под вторым рисунком при нажатии кнопки2 (button2).

22. Дважды щелкните на кнопке btnExit (это кнопка, на которой отображается надпись Exit).

Visual Basic .NET отобразит пустую процедуру обработки событий btnExit.

Наберите в теле процедуры команду Me.Close()

23. Нажмите клавишу <F5> или выберите команду Debug=>Start, чтобы запустить написанную вами программу.

Если вы набрали все коды без ошибок, Visual Basic .NET отобразит на экране пользовательский интерфейс вашей программы.

Тогда при нажатии на соответствующие кнопки под картинками вы увидите примерно следующий рисунок.

Рис.1.2

Не забудьте после создания приложения выполнить команду File=>Save All.

Размеры и тематика рисунков, текстовых полей и кнопок, оформление формы могут быть выбраны самим студентом.

Задание№1.2

Работа с несколькими формами в одном проекте.
Работа с группой переключателей

Постановка задачи. Создать приложение, которое запускается с формы, на которой размещены четыре кнопки:
Кнопка 1 должна быть недоступна для пользователя;
При нажатии **кнопки 2** на ней должен меняться текст.
Кнопка 3 должна открыть другую форму-.Form2, на которой находится надпись “Second Form” с двумя группами переключателей.
Кнопка 4 закрывает приложение.

Указания к выполнению задания

1. Создайте первую форму – Form1.
Разместите на ней с помощью панели инструментов объект Label (надпись). С помощью панели Properties (Свойства) поменяйте свойство Текст: задайте «Задание №2».
Используя панель инструментов, Разместите на ней 4 кнопки.
2. Задайте свойству text следующие значения: для первой кнопки - *недоступен*, для второй - *Кликни сюда,* для третьей – **Форма2,** для третьей – Exit.
Поменяйте фон формы1.
В результате вы получите следующий вид формы, представленный на рисунке 1.3.

Рис.1.3

3. Чтобы сделать **Кнопку 1** недоступной для пользователя, изменить для нее свойство Enabled. Задайте ему значение False. Задайте имя для копки 2 – btnChangeMe.

4. Для того чтобы при нажатии кнопки btnChangeMe на ней менялся текст, дважды щелкните по кнопке btnChangeMe и наберите между началом и концом созданной автоматически процедуры (метода) на событие «Click» (щелчок) команду
btnChangeMe.Text = " I t works!". Поменяйте шрифт текста на кнопке, используя свойство Font.
4. Кнопка 3 должна открыть другую форму-.Form2. Для создания второй формы необходимо проделать следующее:
4.1. Выберите команду Project⇨Add Windows Form (Проект⇨Добавить форму) или щелкните в панели инструментов на кнопке со стрелкой, расположенной справа от значка Add New Item, и выберите пункт Add Windows Form.
Откроется диалоговое окно Add New Item (Добавить новый элемент).
4. 2. Укажите имя новой формы в текстовом поле Name.
Когда вы вводите имя формы, не нужно указывать расширение файла; оно будет добавлено к имени файла автоматически.
4. 3. Щелкните правой кнопкой мыши по новой форме и Visual Basic .NET отобразит новую форму, готовую к применению.
Назовите новую форму **Form2**. Если вы уже знаете, как называется новая форма, которую необходимо отобразить на экране, то щелкнув по кнопке **Форма»**, наберите следующий код BASIC в окне автоматически созданной процедуры на Click:

```
Dim oForm As Form2
    oForm = New Form2()
    oForm. Show ( )
```

4. 4. Создайте на Форме 2 две группы переключателей:
Одна группа для выбора вашей учебной группы из списка групп. Вторая – для выбора предмета, который вы изучаете в данный момент.
Для создания рамки для любой группы переключателей, щелкните на кнопке GroupBox. Курсор мыши примет вид перекрестия. Если вы дважды щелкнете на кнопке инструмента GroupBox, Visual Basic .NET автоматически нарисует рамку группы в окне формы. Потом вам придется переместить ее в нужное место и изменить размеры.
Задайте свойству text объекта GroupBox нужное значение (например, «Выбери группу»). Затем разместите в области GroupBox элемент управления CheckListBox.
Чтобы задать список для переключателей, выделите этот элемент и нажмите на строку **Collections** свойства Items. Щелкните на кнопку с тремя точками, и в открывшемся окне наберите строки для элемента управления CheckListBox.
Проделайте аналогичные действия для другой группы переключателей.
В результате вы увидите форму, представленную на рисунке 1.4:

Рис.1.4

Для выхода из приложения используйте команду Close().

Замечания

1. Возьмите за привычку периодически сохранять результаты проделанной вами работу, тем более что для этого нужно всего лишь нажать комбинацию клавиш <Ctrl+Shift+S> или выбрать команду File⇨Save All. Если ваш компьютер зависнет или если вдруг пропадет напряжение в сети, вы потеряете только ту часть работы, которая была выполнена уже после последнего сохранения.
2. В качестве самостоятельной работы разместите на Form2 два текстовых поля (Textbox1 и Textbpx2), куда запишутся выбранная собственно группа и выбранный предмет.

Для выбранного текстового поля используйте команду

TextBox1.Text = CheckedListBox1.SelectedItem

Задание №1.3

Создание формы для входа в систему с паролем

Постановка задачи. Создать приложение, которое загружается с формы, на которой имеется два текстовых поля: первое - для ввода пароля, другое – для сообщения пользователю о том, верно или неверно введен пароль. Правильный пароль для входа в систему должен быть задан в программе (оператором присваивания). С просьбой повторного ввода (если неверно введен пароль). На форме создать кнопку для подтверждения ввода пароля. Если пароль введен верно, то при нажатии кнопки должно открыться другое окно с сообщением, скажем, «Привет». В противном случае в первом окне во втором текстовом поле выводится сообщение «Пароль неверен».

Виды окна 1 и окна 2 приведены ниже на рисунке 1.5..

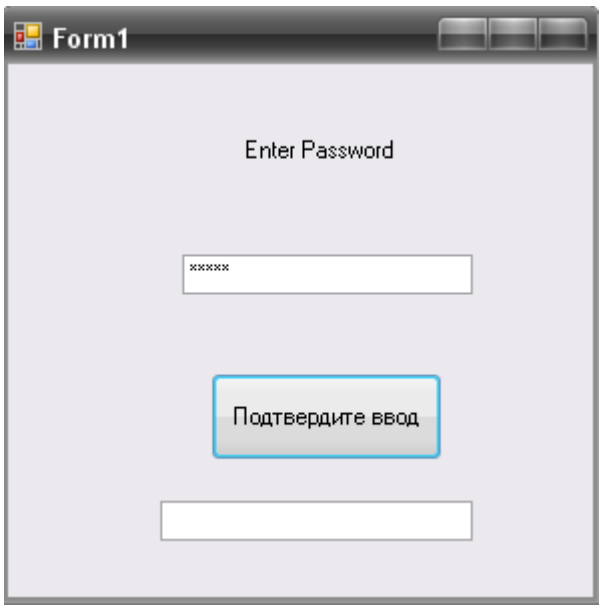

Рис 1.5. Вид окна 1

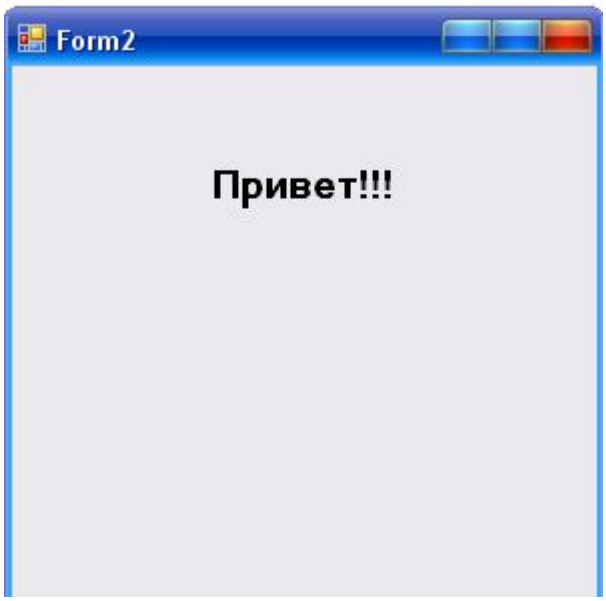

Рис. 1.6. Вид окна 2

Указания к выполнению задания

Создание полей для ввода паролей

Чтобы создать поле для ввода паролей, нужно определить символ, который будет отображаться на экране при наборе любого знака или буквы.

1. Щелкните кнопкой мыши на текстовом поле, которое должно принимать секретные сведения.
2. Откройте окно Properties.
3. Щелкните на свойстве Multiline (категория Behavior).

Рядом с ним появится кнопка со стрелкой, направленной вниз.

4. Щелкните на кнопке со стрелкой и выберите значение False.

Если вы не сделаете этого, т.е. оставите для поля возможность состоять из нескольких строк, скрывать информацию оно не сможет. Таким образом, Visual Basic .NET дает понять, что не стоит создавать огромные пароли.

5. Дважды щелкните на свойстве PassChar (Символ пароля), относящемся к категории Behavior, и наберите какой-нибудь символ (например, звездочку), который будет заменять собой на экране любой введенный знак.
На экран может выводиться только какой-нибудь один символ. Как вы уже могли заметить, наиболее часто для этой цели используется звездочка (*).

5. Создайте на первой форме текстовое поле `TextBox3` и задайте цвет его фона (свойство BackColor) таким же, как и цвет окна формы. Для проверки правильности введенного пароля используйте оператор IF – THEN – ELSE.

```
If TextBox1.Text = TextBox2.Text Then
            Form2.Show()
        Else
            TextBox3.Text = "пароль не верен"
End If
```

Лабораторная работа №2
Работа со списками

Цель: Научиться создавать списки в виде ListBox и ComboBox.и использовать их при решении числовых задач.

Задание

Разработать приложение, которое на основе ввода исходных данных позволит произвести оценочную стоимость дачного дома.

Постановка задачи:

Приложение должно загружаться формы, которая позволит вводить следующие исходные данные:

- длина стены дома;
- ширина стены дома;
- количество этажей;
- материал, из которого строится дом;
- материал, используемый для крыши;
- сведения о том, будет ли у дома мансарда.

Форма должна содержать:

- текстовую информацию пояснительного характера;
- текстовые поля для ввода длины, ширины дома и количества этажей;
- флажок, используемый для указания, будет ли дом содержать мансарду;
- список, позволяющий выбрать кровельный материал;
- раскрывающийся список для выбора материала, из которого будут строиться стены;
- кнопку **Расчет стоимость**, при нажатии на которую будут производиться расчеты стоимости дачного дома;

- поле, размещенное в нижней части формы с правой стороны от кнопки **Расчет стоимости**, предназначено для отображения вычислительной стоимости.
- Кнопку **Сброс,** позволяющую обнулить все числовые данные формы.

Форма для работы с приложением должна выглядеть примерно как на рисунке:

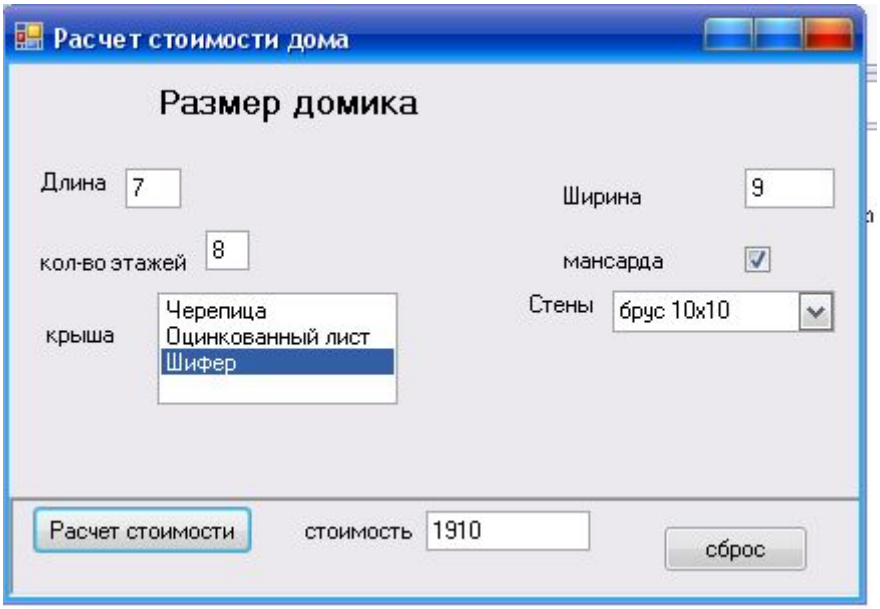

Рис.2

Предполагается, что все четыре стены дома имеют одинаковую площадь. Необходимо предусмотреть невозможность ввода отрицательных значений и символов.

Расчет стоимости дома необходимо производить по следующему алгоритму:

Стоимость стен= 4* длина*ширина*цена_материала* количество этажей.
Стоимость крыши = длина*ширина*цена_кровельного_материала
Стоимость дома= Стоимость стен + Стоимость крыши, если нет мансарды;
Стоимость дома= Стоимость стен + Стоимость крыши + 200, если есть мансарда.

Указания к выполнению задания

1. Создайте новый проект.
2. Создайте форму.
3. Измените имя формы
4. Разместите на форме следующие элементы управления:

- Заголовок «Размер домика» создайте с помощью кнопки Label;
- Текстовые поля для ввода длины, ширины, кол-ва этажей создайте с помощью элемента TextBox;
- Для указания, будет ли домик содержать мансарду, разместите на форме флажок с помощью кнопки CheckBox;

- Для выбора кровельного материала используйте список (кнопка ListBox). Чтобы сформировать список материалов, используйте свойство Items (Collection)
- Для выбора материала, из которого строятся стены, используйте раскрывающийся список (кнопка ComboBox). Для ввода списка воспользуйтесь свойством Items (Collection).
- Создайте кнопку для расчета стоимости (кнопка **Button1**)
- Создайте кнопку **Сброс** для обнуления всех текстовых полей, а также сброса флажка (кнопка Button)

5. Напишите процедуры обработки события **Click** для каждой кнопки.

Наличие поставленного флага проверяется свойством CheckBox.Checked (=true или false)

6. Цена на каждый вид материала задается в процедуре обработки события Click_Button1.

7. Выбранный вид проверяется по значению свойства `ListBox1.SelectedIndex`. Причем индексация начинается с нуля.

Все переменные объявляются целыми.

8. Ниже представлен фрагмент программы расчета домика без учета расходов на крышу.

```
Dim l, h, store, cena, stoim, mans As Integer
        mans = 0
        l = TextBox1.Text
        h = TextBox2.Text
        store = TextBox4.Text
        If CheckBox1.Checked = True Then mans = 20
        If ListBox1.SelectedIndex = 0 Then cena = 10
        If ListBox1.SelectedIndex = 1 Then cena = 20
        If ListBox1.SelectedIndex = 2 Then cena = 30
        stoim = 4 * l * h * cena + mans
          TextBox3.Text = stoim
```

9. Факт установления флажка соответствует тому, что значение `CheckBox1.Checked` равно `True`.

10. Ниже показан код программы на событие – изменение значения в textbox1 (проверка на ввод только числовых значений):

```
Private Sub TextBox1_TextChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
TextBox1.TextChanged
        If IsNumeric(TextBox1.Text) = False Then
            TextBox1.Text = "repeat input"
        End If
    End Sub
```

Замечание. Необходимо сделать проверку при вводе в текстовые поля на положительность числовых значений (при вводе других значений должно выводится сообщение об ошибке).

Лабораторная работа №3
Работа с классом математических функций.

Цель: Научиться работать со стандартными математическими функциями в VB.NET.

Создание проекта Калькулятор

Постановка задачи

Создать форму, на которой будут иметь место следующие элементы:

1. Два поля с именами Операнд 1 и Операнд 2, предназначенные для ввода операндов;
2. Поле для отображения выполняемой операции;
3. Поле для результата;
4. Командные кнопки со стандартным для калькуляторов назначением;
5. Кнопка "Выход" для окончания работы.

Приведем *сценарий работы пользователя с калькулятором*

В полях ввода вводится необходимое для операции число операндов, после чего кликом по командной кнопке выполняется требуемая команда, отображаемая в поле оператора; в поле результата читается результат операции. Если результат должен, далее, участвовать как операнд в следующей операции, то кликом мыши в поле результата мы копируем его в поле первого операнда. Для быстрой очистки полей ввода можно использовать двойной клик.

Аргументом для тригонометрической операции должен быть угол в градусах (программа должна переводить градусы в радианы).

В случае ввода нечисловых данных, деления на ноль, вычисления корня квадратного из отрицательного числа должно выводиться дополнительное окно с сообщением об ошибке.

Пользователь программы может вводить дробные числа.

Примечание. В качестве разделителя целой части от дробной используйте знак “,”.

Указания к выполнению задания

1. Создание формы

Создать форму, представленную на рисунке 3.

2. Ввод операндов

Если мы установим курсор в поле ввода (возьмем этот объект тем самым "в фокус") и начнем набор символов на клавиатуре, то набираемая строка на каждом шаге набора будет значением свойства Text ("текст") поля ввода, и в качестве этого значения будет в поле ввода отображаться.

«Подцепление символа» производится с помощью одной из двух операций: "+" или "&" (перед значком "&" - "амперсандом" - нужно обязательно самим ставить пробел!). Таким образом, обработчик может быть таков:

```
TextBox.Text = TextBox.Text & Button2.Text 'Приписываем к примеру цифру 1,
или
TextBox.Text = TextBox.Text & Button1.Text  'Приписываем точку
```

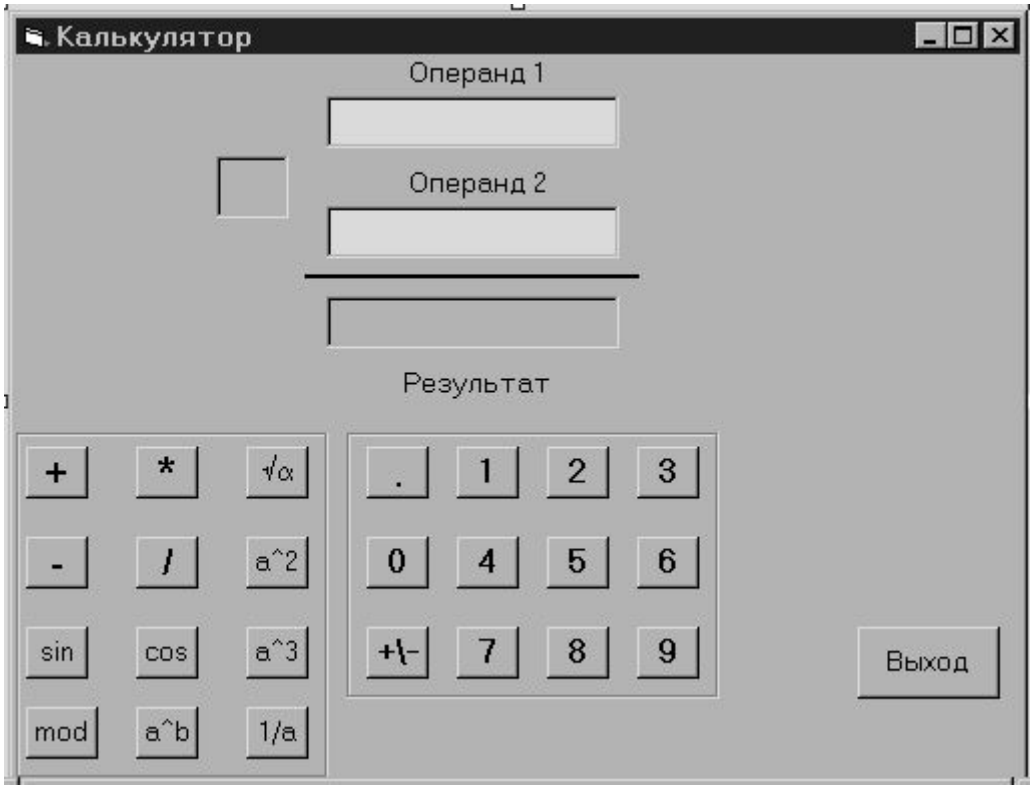

Рис.3

Взять объект в фокус в VB означает вызвать обработчик TextBox2_GotFocus.
Совет: Можно использовать некоторую глобальную переменную, например i (integer), хранящей индекс (1 или 2) поля ввода, бывшего в фокусе последним. Начальное присвоение (i=0) делается обычно при загрузке формы (событие Load).
При реализации операции деления необходимо предусмотреть невозможность деления на ноль.

2. Вычисление результатов операций

Осуществляется при нажатии на каждую из операционных кнопок (событие Click)
Используемые функции VB для реализации в процедурах:
Корень квадратный – **Sqrt**
Остаток от деления – **Mod**
Возведение в степень – знак **“^”** и т.д.
Для работы с математическими функциями надо вызвать соответствующую математическую функцию класса Math. Например, для вычисления корня квадратного из числа t, необходимо записать Math.Sqrt(t).
При работе с тригонометрическими функциями не стоит забывать, что аргумент, введенный в градусах, необходимо перевести в радианы умножением на 3.14/180.

3. Вывод результатов

Вывод осуществляется в поле Результат и в ярлыке результата для бинарных операций (сложение, вычитание, умножение, деление, возведение в степень, остаток от деления).

Лабораторная работа №4
Создание MDI интерфейса. Создание меню.
Работа с диалоговыми окнами

Цель: Научиться работать с дочерними окнами и создавать меню.

Постановка задачи
Создать приложение, которое загружается с окна, представленного на рис.4.1.

Рис.4.1. Вид пользовательского интерфейса

1. Создайте родительскую и дочернюю форму.
2. Создайте меню и панель инструментов для обеих форм.
3. Создайте два пункта меню верхнего уровня с именем mnuFile и mnuWindow и текстом File и Window, соответственно.
4. Создайте три пункта меню File со следующими свойствами:

Таблица 1.

Name	Text	ShortcutKeys
mnuFileOpen	Open	
mnuFileSave	Save	<Ctrl>+<S>
mnuFileExit	Exit	<F10>

5. Меню Open должно иметь 2 подменю New и Existing.
   Подменю New открывает новое дочернее окно и имеет горячие клавиши <Ctrl>+<N>.
   Подменю Existing служит для поиска в файловой структуре нужного файла.
   Подменю Save служит для выбора места хранения файла и ввода его нового имени.
6. В меню Windows должен отображаться список открытых дочерних окон (Рис.4.2).

7. Создайте строку состояния, показывающую текст подсказки, текущую дату и время.
8. Создать окно сообщений для закрытия приложения, открытия файла, сохранения файла.
9. При выборе пункта Open должнен открываться файл в пользовательской форме.
10. Создайте панель инструментов.

Рис.4.2. Отображение открытых окон.

Указания к выполнению:

1. Создание родительской и дочерней формы.

Родительское окно MDI – интерфейса является контейнером для всех дочерних окон.

Для создания родительского окна необходимо присвоить значение **True** свойству IsMdiContainer стандартной формы Windows. При задании значения фон формы потемнеет.

Дочернее окно может находиться только внутри родительского окна. При разворачивании дочерние окна занимают все пространство родительского окна, а к его заголовку добавляется заголовок активного дочернего окна в квадратных скобках. При сворачивании дочернего окна его пиктограмма отображается внутри родительского окна.

Для создания дочернего окна MDI – интерфейса необходимо выполнить следующее:

1.1.Добавьте в проект еще одну форму. Задайте ей заголовок **Дочерняя форма** и имя frmChildMdi.

1.2. Запрограммируйте вызов дочерней формы при открытии родительского окна. Для этого выделите родительскую форму в конструкторе форм и с помощью команды ViewCode вызовите редактор родительского окна.

1.3. Добавьте в код следующую процедуру:

```
Public Class Form1
```

```
Private Sub NewChildForm()
    Dim f As New frmChildMdi
    f.MdiParent = Me
    f.Show()
  End Sub
```

Чтобы дочернее окно открывалось при запуске родительского окна, добавьте в конструктор или метод Load родительского окна строку:

```
NewChildForm()
```

После задания имени родительского окна свойство IsMdiChild дочернего окна автоматически принимает значение True.

Расположение дочерних окон в родительском окне можно управлять при помощи метода LayoutMdi.

2. Создание меню для MDI – интерфейса.

2.1. Перетащите на форму родительского окна элемент управления MenuScrip и присвойте ему имя mnuParent. Создайте два пункта меню верхнего уровня с именем mnuFile, mnuWindow и текстом File и Window, соответственно.

2.2. Для того чтобы в меню Window отображался список дочерних окон, присвойте значение MdiWindowListItem элемента управления mnuParent.

2.3. Создайте пункты для меню File со свойства, представленными в таблице 3.

2.4. Перетащите на форму дочернего окна элемент управления MenuScrip и присвойте его свойствам Name, Visible значения mnuChild и False соответственно. Создайте для него аналогичный пункт меню File и задайте для него свойства MergeAction значение MatchOnly. Добавьте в данный пункт меню команду с именем mnuFileClose и текстом Close File.

2.5. Создайте обработку события выбора команды New меню File. Для этого в обработчик события добавьте NewChildForm()

2.6. Перенесите элемент PictureBox на форму. Создайте диалоговое окно открытия файла, которое служит для поиска файлов, используемых в программе.

Для создания этого окна предназначен элемент управления OpenFileDialog. Чтобы отобразить диалоговое окно открытия файла, необходимо перенести на форму элемента управления OpenFileDialog вызвать его метод ShowDialog. С помощью метода OpenFile данного элемента можно открыть поток для чтения данных из файла.

Решим следующую задачу: откроем окно и отобразим содержимое диска C. При этом должны отображаться лишь графические объекты.

Для этого необходимо написать в процедуре обработки события вызова пункта Existing подменю Open из меню File следующий код:

```
OpenFileDialog1.Filter = "Image
Files(*.BMP;*JPG;*.Gif)|*.BMP;*JPG;*.Gif"
        OpenFileDialog1.InitialDirectory = "c:\"
          OpenFileDialog1.ShowDialog()
```

Для того чтобы отразить содержимое картинки открытого файла в созданном на форме объекте PictureBox, необходимо в коде прописать команды: PictureBox1.image=Image.FromFile(OpenFileDialog1.FileName).

2.7. Для поиска файла, в котором будут сохранены данные из программы, используется диалоговое окно сохранения файла. Для создания этого окна служит элемент управления SaveFileDialog, работа с которым аналогична работе с элементом OpenFileDialog. С помощью метода OpenFile данного элемента можно открыть поток не только для чтения данных из файла, но и для записи.

Код обработки события вызова пункта меню Save:

```
SaveFileDialog1.Filter = "All Files(*.*)|*.*"
        SaveFileDialog1.ShowDialog()
        If SaveFileDialog1.FileName > "" Then
            MessageBox.Show("Выбран файл с именем" +
SaveFileDialog1.FileName)
          End If
```

2.8. Для создания окна сообщения, позволяющего завершить приложение, добавьте в код программы процедуру обработки события вызова пункта Exit, дважды кликнув по этому пункту меню. Внесите в эту процедуру следующий код:

```
Private Sub ВыходToolStripMenuItem_Click(ByVal sender
As Object, ByVal e As System.EventArgs) Handles
ВыходToolStripMenuItem.Click
Dim result As DialogResult
    result = MessageBox.Show("Вы действительно хотите
закрыть приложение?", _
       "Закрытие приложения", MessageBoxButtons.YesNo,
MessageBoxIcon.Asterisk)
      If result = DialogResult.Yes Then
          Application.Exit()
        End If
End Sub
```

2.9. Создание панели инструментов.

Чтобы создать панель инструментов, выполните следующие действия:

- В родительскую форму добавьте элемент управления ToolScrip.
- Для размещения кнопок на панели инструментов откройте диалоговое окно **ItemsCollectionEditor**, выбрав свойство Items панели.
- В раскрывающемся списке **Select item and add to list below** выберите элемент Button. С помощью кнопки **Add** добавьте элемент с именем tsbnew. Данная кнопка будет создавать новое дочернее окно.
- Задайте текст подсказки, выводимый при задержке указателя мыши на кнопке tsbNew, присвоив значение **Создание новой дочерней формы** свойству ToolTipText.
- Расположите на кнопке изображение с помощью свойства Image.

- Теперь задайте обработку события нажатия на кнопку. Для этого добавьте в процедуру обработки события tsStandart_ItemClick следующий код:

```
Select Case ToolStrip1.Items.IndexOf(e.ClickedItem)
    Case 0
        NewChildForm()
End Select
```

2.10. Добавим в приложение MdiExample строку состояния, показывающую текст
подсказки, текущие дату и время. Для этого откройте приложение и выполните следующие действия:

- Добавьте в родительскую форму элемент управления StatusStrip, дважды щелкнув мышью кнопку StatusStrip (Строка состояния) lifci панели инструментов. После появления в форме строки состояния присвойте ей имя ssStatusBar.
- Откройте окно свойств Properties (Свойства) строки состояния и выберите свойство Items. Откроется диалоговое окно Items Collection Editor (Редактор списка элементов), в котором с помощью раскрывающегося списка Select item and add to list below (Выберите элемент и добавьте в нижерасположенный список) и кнопки Add (Добавить) добавьте в строку состояния три элемента StatusLabel и назовите их ssrText, ssrDate и ssrTime.
- Для свойства Text панели ssrText задайте значение Текст подсказки.
- Для того чтобы время и дата постоянно обновлялись при работе приложения, перетащите на родительскую форму элемент управления Timer и назовите его timer.
- Затем щелкните дважды на этом элементе, чтобы открылось окно программного кода на процедуре обработки события timer_Tick. Добавьте в данную процедуру следующий код, который позволит при каждой смене значения таймера обновлять данные строки состояния:

```
ssrDate.Text = System.DateTime.Today.ToLongDateString
ssrTime.Text = System.DateTime.Now.ToLongTimeString
```

- Для активизации таймера добавьте в процедуру обработки Load родительского окна строку
  timer.Enabled = True
- Для дезактивизации таймера в процедуру обработки события `ВыходToolStripMenuItem_Click` добавьте следующую строку:
  timer.Enabled = False

Лабораторная работа №5
Работа с массивами: поиск, сортировка.
Работа с глобальными переменными

Цель: Научиться работать с массивами, делать поиск элементов в массиве и проводить сортировку, создавать глобальные переменные.

Постановка задачи.

Разработать приложение, реализующее сортировку и поиск в заданном массиве.

Приложение должно вызывать последовательно две формы. На первой форме имеется две кнопки для вызова двух форм: первая кнопка – для проведения сортировки, вторая для поиска нужного элемента в массиве (Рис5).

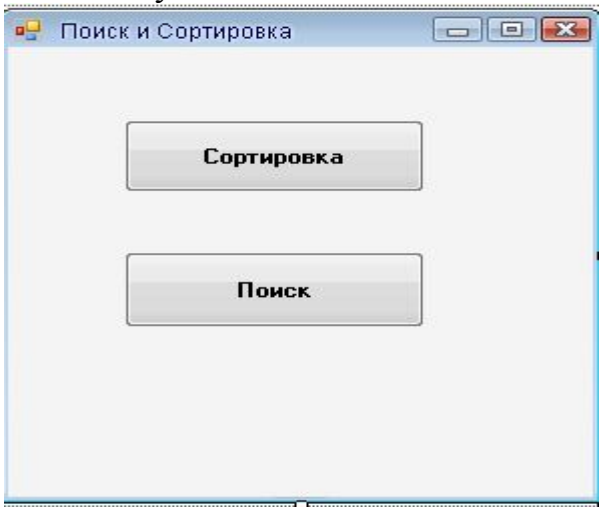

Рис.5.1. Главная форма

При нажатии первой кнопки открывается форма для ввода неотсортированного массива через Rich textbox. Результат также должен быть записан в текстовое поле (Рис.5.2).

Рис.5.2. Окно сортировки

Для поиска элемента массива и для вывода результата использовать текстовые поля (Рис.5.3).

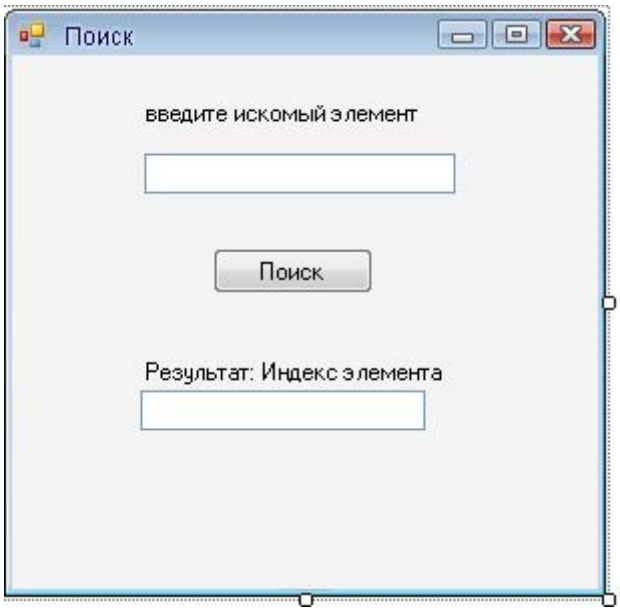

Рис.5.3. Окно поиска.

Операции по вводу массива делать в первой форме. Массив может быть как символьный, так и числовой. Единственное требование к массиву – он должен быть объявлен глобальным.

Указания к выполнению:

1. Создать проект. В проекте создать модуль для объявления сортируемого массива и результата поиска номера искомого элемента глобальными.
   - Для создания модуля следует щелкнуть правой кнопкой в окне Solution Explorer и выполнить команду Add, затем выбрать строку Module…
   - В открывшемся окне наберите

```
Module Module1
    Public i
    Public strnames() As String
End Module
```

2. Создать форму1 и поместить туда кнопки для выбора задачи: осуществления поиска или сортировки объявленного в модуле массива (Рис.6)
3. Создать форму2 для вывода результатов сортировки. Для чтения элементов массива из RichTextBox1 используйте свойство RichTextBox1.Lines(i), а для записи в RichTextBox2 результатов сортировки массива используйте цикл вида:

```
For i = 0 To 9
        RichTextBox2.Text = RichTextBox2.Text &
strnames(i) & Chr(13)
Next i
```

4. Сортировку осуществлять с помощью команды Array.Sort(<имя массива>).

5. Создать форму3 для проведения поиска (Рис.7). Поиск осуществляется с помощью команды

i=Array.BinarySearch (<имя массива>, <элемент массива>).

Если элемент найден, то возвращается индекс, иначе – отрицательное число. При этом после появления формы3, форма2 должна исчезнуть (Form2.hide). Если элементы массива не введены, то должно быть выведено соответствующее сообщение.

Задание для самостоятельной работы:

Написать приложение, в котором исходный массив считывается из файла.

Лабораторная работа №6
Реализация численных методов

Цель: Научиться создавать приложения на VB.Net, реализующие численные методы.

Задание 6.1

Разработать приложение, реализующее численные методы вычисления интеграла: прямоугольников и Симпсона. Интервал интегрирования и вид функции задаются индивидуально каждому студенту из таблицы 2 (интервал задается значениями a и b, f(x)-подынтегральная функция). Точность задать самостоятельно.

Задание 6.2

Разработать приложение, реализующее поиск оптимального значение функции методом «золотого сечения». Для выполнения заданий каждому студенту задается функция из таблицы 3.

Задание 6.3

Разработать приложение, реализующее по выбору пользователя один из численных методов решения нелинейного алгебраического уравнения: хорд, касательных (Ньютона), простой итерации, дихотомии (деления отрезка пополам). Варианты уравнений каждому студенту задаются индивидуально.

Таблица 2

Варианты для выполнения задания 6.1

№п/п	Функция f(x)	a	b	№п/п	Функция f(x)	a	b
1	$x^3e^{Sin(\pi x)}$	0	2	15	$x^5e^{-x/Sin(x)}$	0,5	2
2	$x^3e^{Cos(x)}$	0	2	16	$2Cos(Cos(x))$	-1	1
3	$3Sin(Sin(x))$	0	2	17	$(1-x)^5e^{xSin(x)}$	0	2

4	$e^x Ln(1+x)$	0,5	1,5	18	$3Cos(Cos(x))$	-1	1
5	$2Sin(Sin(x))$	0	3	19	$(1+x)Ln(1+x)$	0	2
6	$10^{Ln(1+x)}Sin(x)$	0	2	20	$x^{xLn(x)}Ln(1+x)$	0,5	1,5
7	$e^x Ln(1+x)$	0	1	21	$x^2 Ln(x)e^x$	0,5	1,5
8	$Ln^2 x - Cos(x)$	1	2	22	$Sin(Ln(1+x))e^x$	0	2
9	$xLn(x)$	0,5	2	23	$Cos(x)Ln(x)$	1	3
10	$Cos(Ln(1+x))e^x$	0	1,5	24	$x^3 - xCos(\pi x)$	0	1
11	$e^x Sin(x)$	0	1	25	$Cos(Tg(x))e^x$	0	1
12	$Sin(x)(1+x^3)$	0	2	26	$Cos(\pi x^2)$	0	2
13	$Cos(1+\pi x^2)$	0	2	27	$Ln(\pi x - x^2)$	1	2
14	$Ln(\lvert 1-x^3 \rvert)e^x$	1	2	28	$Sin(x)+Cos(x)$	0	

Содержание работы №6.1.

1. Разработать блок-схемы работы алгоритмов для вычисления определённого интеграла методом прямоугольников и Симпсона.
2. На рисунках 6.2 и 6.3 представлены блок-схемы алгоритмов для вычисления определённого интеграла методами прямоугольников и Симпсона, где a, b – границы интегрирования, e – заданная точность.
3. Разработать интерфейс пользователя.

В интерфейсе задаются границы интегрирования, точность вычисления, возможность выбора метода реализации (можно использовать CheckBox), текстовое поле для результата. На рис. 6.1 представлено окно загружаемой формы (примерный вид):

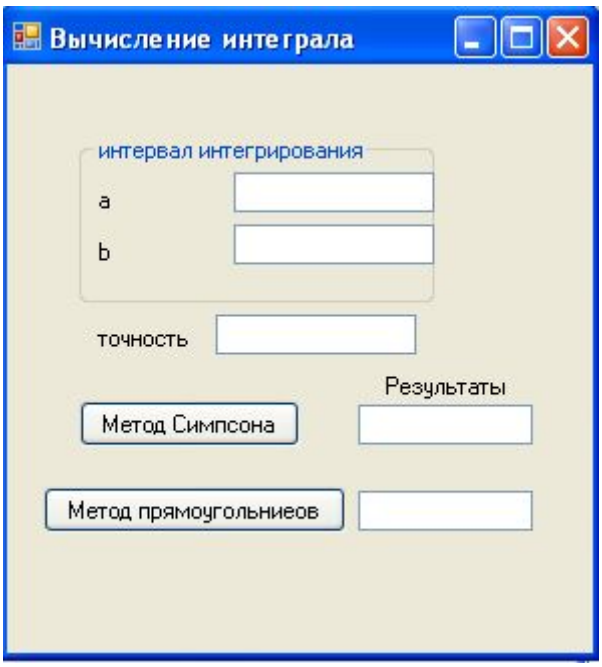

Рис.6.1.

Для проверки правильности работы приложения вычислите определенный интеграл вручную.

Задание для самостоятельной работы:

1. Самостоятельно разработать алгоритм вычисления интеграла методом трапеций и добавить соответствующую ему процедуру в приложение.

Содержание работы №6.2

1. В MS Excel построить графики заданных функций, чтобы найти диапазон, в котором будет искаться оптимальное значение функции.
2. Разработать процедуру, реализующую алгоритм поиска экстремального значения функции с использованием метода золотого сечения. Алгоритм поиска экстремумов методом золотого сечения представлен на рис.6.4.
3. В пользовательском интерфейсе должны быть текстовые поля для ввода исходных данных, вывода результата, показа оптимизируемой функции.
4. Варианты заданий оптимизируемой функции приведены в таблице 3.
5. Решение провести с различными значениями точности и сравнить результаты.

Таблица 3

Варианты заданий к лабораторной работе № 6.2

№	Функция f(x)	a	b	№	Функция f(x)	a	b
1	$\cos(x+\pi/4)/\sqrt{x}$	6	8	15	$2\cos(2x)-4$	5	6
2	$\sin(x+\pi/2)/\sqrt{x}$	7	9	16	$\cos(\pi x/2)/(1-x)$	0	2

3	$\sin(x+3\pi/4)/\sqrt{x}$	9	11	17	$\sin(\pi x)/x^{1+x}$	0	1
4	$\cos(x+\pi/4)/\sqrt{x}$	7	9	18	$\sin(x+\pi/4)/\sqrt{x}$	1	3
5	$\sin(x)+5\sin(3x)$	2	3	19	$\sin(x+3\pi/4)/\sqrt{x}$	3	5
6	$3\sin(x)+\sin(3x)$	0	2	20	$\cos(x+3\pi/4)/\sqrt{x}$	1	3
7	$\cos(x)-\cos(3x)$	4	6	21	$(1-\cos(x))/\sqrt{x}$	1	3
8	$x^2\ln(x)$	2	3	22	$\sin(x)+5\sin(3x)$	0	1
9	$\ln(1+x)\sin(x)/x^2$	7	9	23	$3\sin(x)+\sin(3x)$	7	9
10	$\ln(x)\cos(x)x^2$	2	4	24	$\cos(x)-\cos(3x)$	0	2
11	$\ln^2(x)-\cos(x+1)$	4	5	25	$x\ln(1+x)e^x$	1	2
12	$\sin(\ln(1+x))e^x$	0	2	26	$\ln(1+x)\sin(x)/x$	1	3
13	$\ln(\pi x)/x$	1	3	27	$\ln(x)xe^x$	2	3
14	$E^x\ln(1+x^2)$	1	3	28	$\cos(\ln(1+x))e^x$	0	3

Содержание работы №6.3

1. Разработать блок-схему работы одного из алгоритмов для решения нелинейного алгебраического уравнения. Алгоритм поиска корня методом дихотомии (делением отрезка пополам) представлен на рис.6.5.
2. Разработать приложение, реализующее выбранный метод. В пользовательском интерфейсе должны быть текстовые поля для ввода исходных данных, вывода результата, показа оптимизируемой функции.
3 . Варианты заданий оптимизируемой функции и интервалы поиска приведены в таблице 6.3.
4. Построить в EXCEL график функции, чтобы убедиться, что для заданной функции в заданном диапазоне имеется оптимум.

Таблица 3

Варианты уравнений к лабораторной работе № 6.3

№	Функция f(x)	a	b	№	Функция f(x)	a	b
1	Ln(x)=1/x	1	2	15	1/ex=x2	0	1
2	Ln(x)=Sin(x)	1	3	16	x+x3=5	1	2
3	Sin(x)=1/x	0	x/2	17	Ln(x)=Sin2(x)	0	x/2
4	Sin(x)=x/2	x/2	x	18	Ln(x)=1/ex	0	2
5	Cos(x)=x	2	5	19	Lg(x)=1/ex	1	3
6	Cos(x)=Ln(x)	0	x/2	20	Cos(x)=x3	0	x/2
7	Cos(x)=Tg(x)	0	x/2	21	Cos(x)=x2	0	x/2
8	Cos(x)=1/x	4	6	22	Lg(x)=1/10x	7	9
9	Cos(x)=Ln(1+x)	0	x/2	23	Tg(x)=1/x	0	2
10	Sin(x)=x/3	x/2	x	24	Ln(1+x)/x=2/π	0,1	2
11	1/ex=x	0	1	25	2+Ln(x)=1/x	0,1	1
12	Ln(x)=1/x2	1	2	26	2+Ln(x)=1/x2	0,1	1
13	1/ex=Sin(x)	1	2	27	Tg(x)=1/x2	0	x/2
14	ex=1/Sin(x)	1	2	28	Tg(x)=1/x	0	x/2

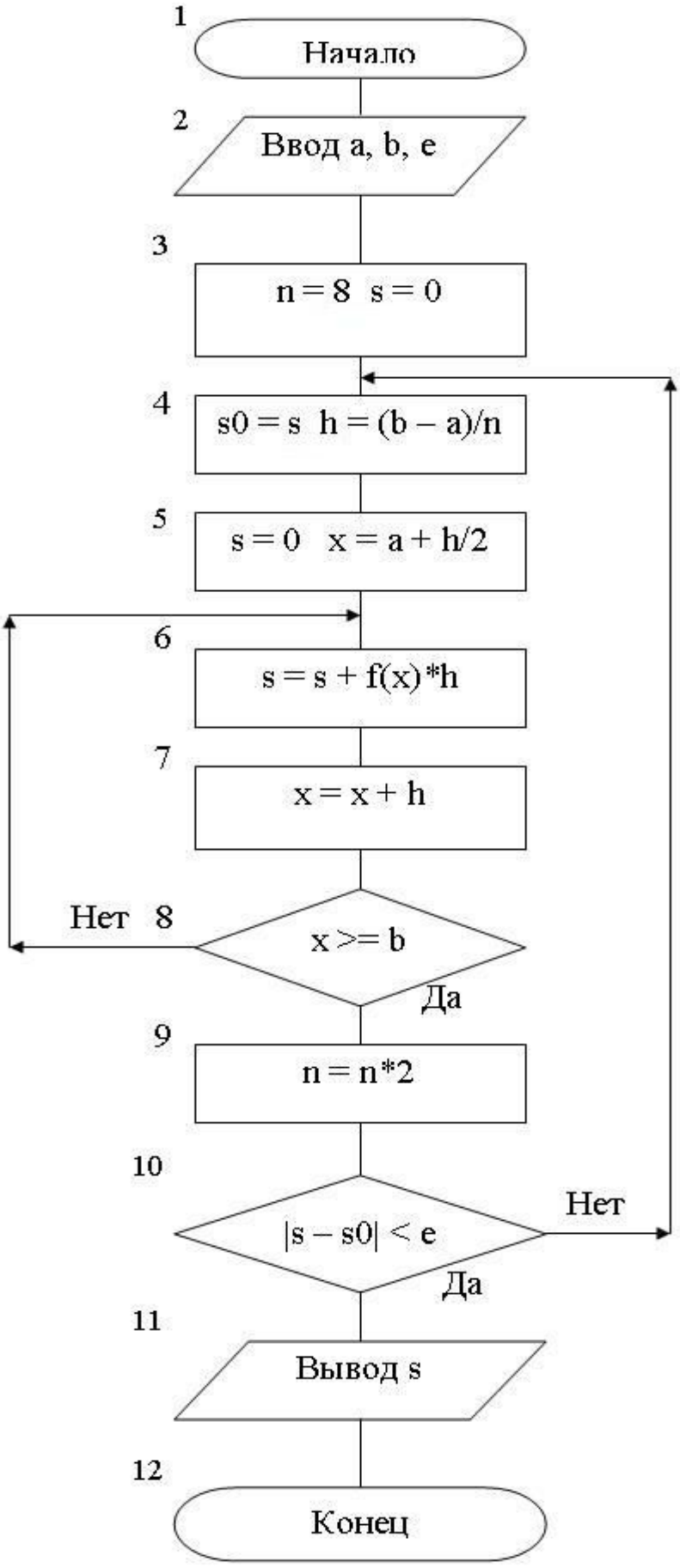

Рис.6.2. Метод прямоугольников

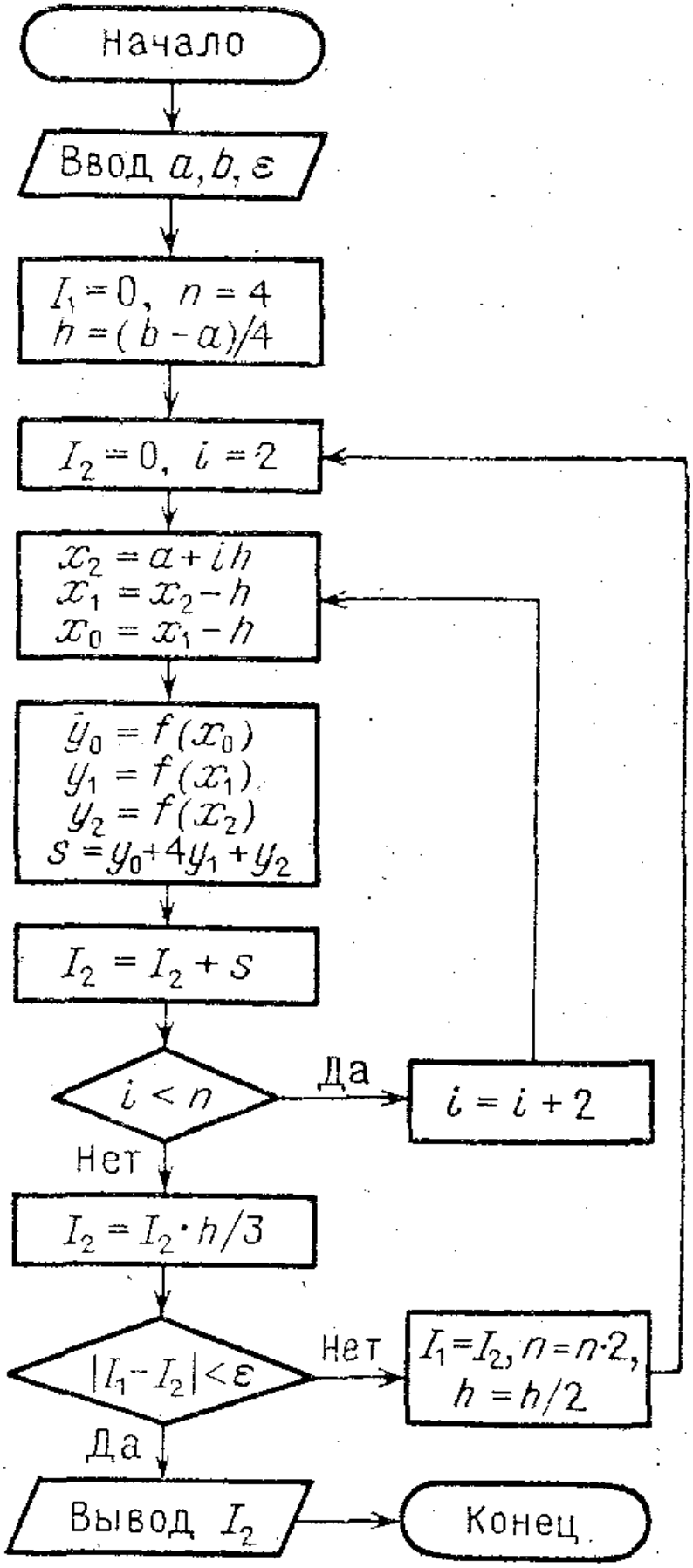

Рис.6.3. Метод Симпсона

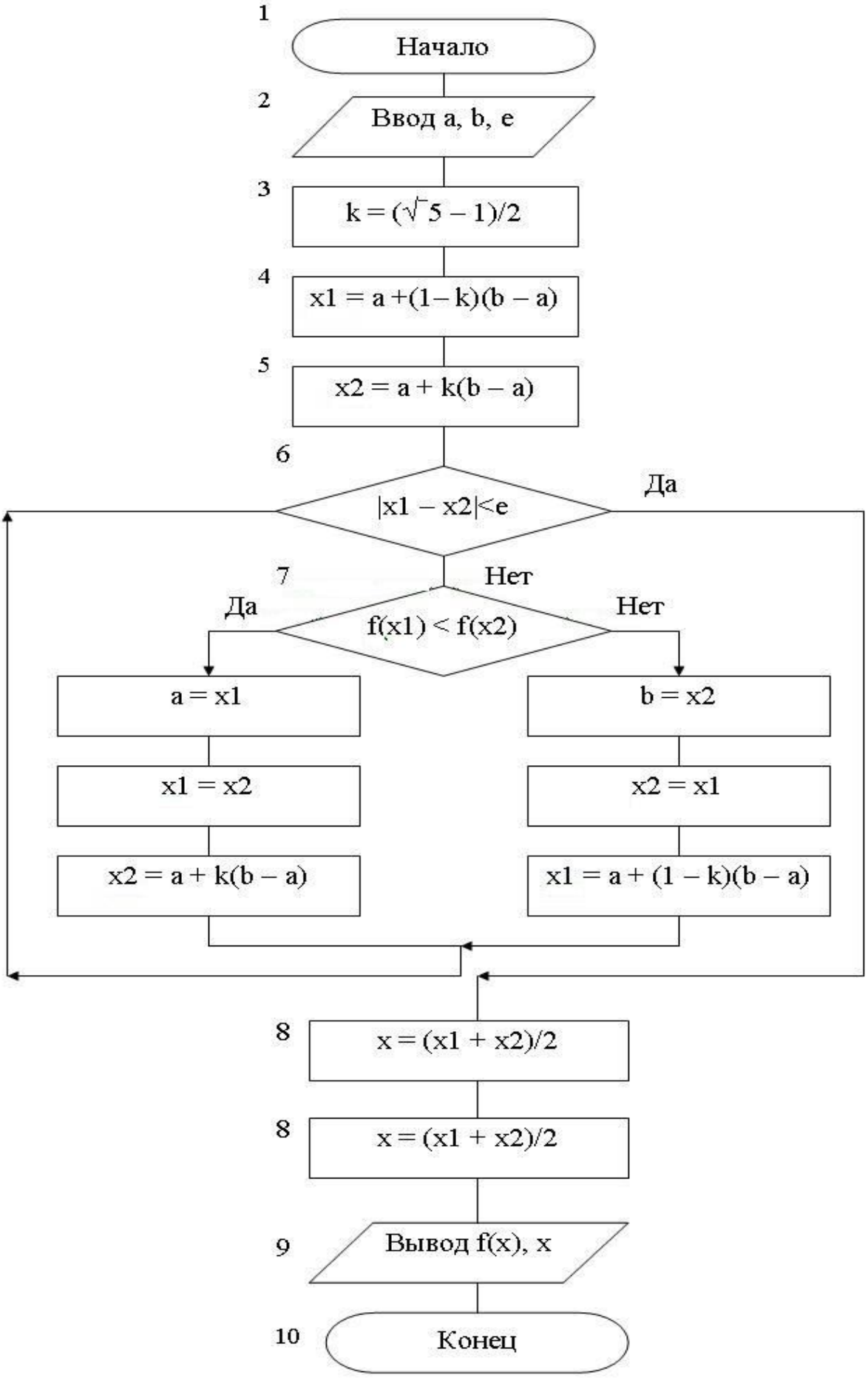

Рис.6.4. Метод золотого сечения

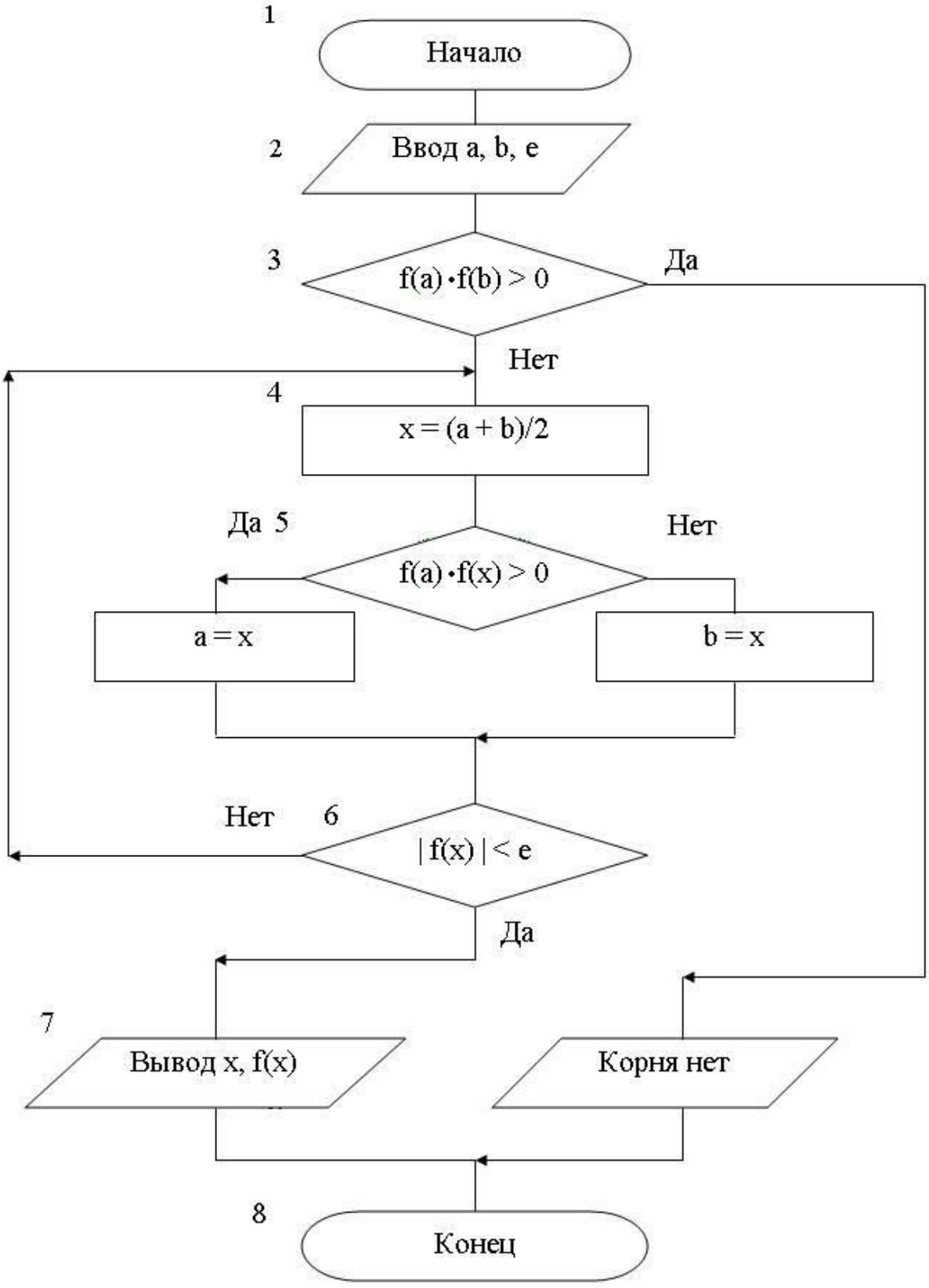

Рис.6.5. Метод дихотомии

Лабораторная работа №7
Построение графиков функций

Цель: Научиться строить графики в VB.Net.

Постановка задачи

Разработать приложение для построения графика функции вида $\mathbf{Y=ax^3+bx^2+cx+d}$ на заданном диапазоне.

Коэффициенты функции должны вводиться с помощью текстовых полей. График в видео окно выводится нажатием на кнопку (Button1).

Возможный вариант формы представлен на рис.7.1

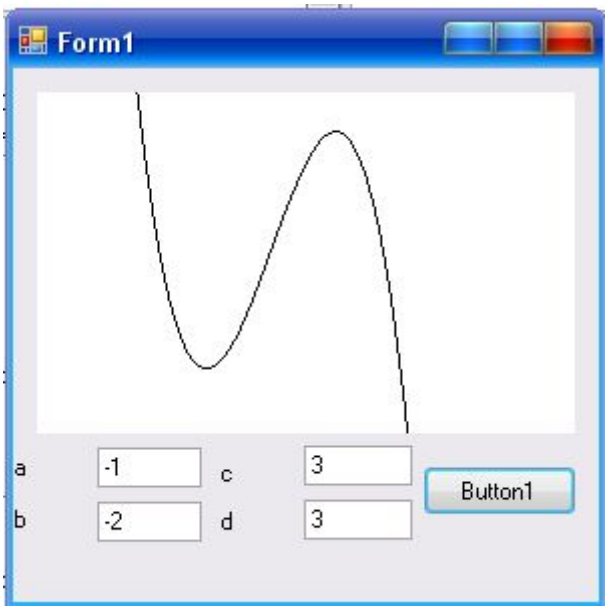

Рис.7.1

Методика выполнения задания

Для построения графика функции мы должны решить несколько проблем (подзадач):

- На экране необходимо отобразить ось Y, направленную вверх, тогда как по умолчанию изменение пикселей по оси Y направлено вниз. Поэтому мы должны различать между координатами x, y и их значениями в пикселях – xpixel, ypixel.
- Затем мы должны гарантировать, что график будет эффективно использовать приготовленный для него PictureBox (будет правильно смасштабирован). Мы предполагаем, что доступная область - 200 пикселей по оси x и 200 пикселей по оси y. В программе будем считать, что показать x и y изменяются в диапазоне -5.0 к +5.0. Таким образом, 1 единица x (или y) составляет 20 пикселей.
- Наконец, так как мы будем использовать DrawLine, чтобы представить график, мы вынуждены нарисовать его в виде большого количества маленьких отрезков. Мы будем двигаться по направлению x, один пиксел за

один шаг, прочерчивая линию от одной соответствующей y-координаты до следующей. Для каждого x пиксела, программа выполняет следующее:

1. Вычисляет значение x по значению x пикселей;
2. вычисляет значение y как функцию (по формуле);
3. вычисляет значение y пикселей по значению y, используя следующие команды:

x=ScaleX(xpixel)
y=TheFunction(x)
ypixel=ScaleY(u)

Затем программа переходит к следующему x пикселю и вычисляет снова соответствующий ypixel:

Nextxpixel=xpixel+1
Nextx=Scalex(nextxpixel)
Nexty=TheFunction(nextx)
Nextypixel=Scale(nexty)

Наконец рисуется маленький отрезок ломанной линии: **Paper.Drawnline(myPen,xpixel,ypixel,nextXpixel,nextYpixel)**.

Приведем полностью код программы рисования графика.

```
'Объявляем переменные уровня класса:
Private a, b, c, d As Double
Private paper As Graphics
Private mypen As Pen = New Pen(Color.Black)
' Прописываем методы:
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
      paper = PictureBox1.CreateGraphics()
      DrawGraph()
   End Sub

Private Sub DrawGraph ( )
      a = cdbl(textbox1.text)
      b = cdbl(textbox2.text)
      c = cdbl(textbox3.text)
      d = cdbl(textbox4.text)
      paper.Clear(Color.White)
      Draw( )
End Sub

Private Sub Draw()
Dim x, y, nextX, nextY As Double
Dim xpixel, ypixel, nextXPixeL, nextYPixel As Integer
For xPixel = 0 To PictureBox1.Width
      X = ScaleX(xPixel)
      y = TheFunction(x)
```

```
        yPixel = ScaleY(y)
        nextXPixel = xPixel + 1
        nextX = Scalex(nextXPixe1)
        nextY = TheFunction(nextX)
        nextYPixeL = ScaleY(nextY)
        paper.DrawLine(myPen, xpixel, _
              yPixel, nextXPixe1, nextYPixel)
'paper.DrawLine(mypen, xpixel, _
'             100, nextXPixeL, 100)
'             paper.DrawLine(mypen, 100, _
'              ypixel, 100, nextYPixel)
next
End Sub

Private Function TheFunction (ByVal  x As Double) As Double
Return a * x * x * x + b * x * x + c * x + d
End Function

Private Function ScaleX(ByVal xPixel As Integer) As Double
Dim xStart As Double = -5, xEnd AS Double = 5
Dim xScale As Double = PictureBox1.Width / (xEnd - xStart)
Return (xPixel - (PictureBoxl.Width / 2)) / xScale
End Function

Private Function ScaleY(ByVal y As Double) AS Integer
        Dim yStart As Double = -5, yEnd As Double = 5
        Dim pixelCoord As Integer
        Dim yScale As Double = PictureBoxl.Height / (yEnd - yStart)
        pixelcoord = CInt(-Y*yScale) + CInt(PictureBoxl.Height / 2)
Return pixelCoord
End Function
```

Задание для самостоятельной работы:

Добавьте на диаграмме горизонтальную и вертикальную оси координат, используя метод DrawLines класса Graphics, и числа по обеим осям, используя метод DrawString того же класса.

Лабораторная работа №8
Работа с текстовыми файлами

Цель: Научиться записывать информацию в текстовые файлы и считывать ее.

Общие теоретические сведения

При работе с файлами в Visual Basic 2008 применяется пространство имен System.IO, классы которого позволяют создавать, копировать, перемещать, удалять файлы и каталоги, а также многое другое. Важными операциями при работе с файловыми потоками является чтение и запись данных. Для чтения текстовых данных из файла и записи в него новой информации в пространстве имен System.IO существуют классы StreamReader и StreamWriter.

Для считывания отдельных строк файла предназначен метод ReadLine класса StreamReader, а посимвольно – метод Read. Аналогично, для записи отдельных строк файла предназначен метод WriteLine, а посимвольно – метод Write.

Постановка задачи.

Разработать приложение для считывания и записи информации в текстовые файлы:

Приложение должно предоставлять пользователю следующие возможности:

1. Считывать из текстового файла информацию построчно и посимвольно (распознавать элементы массива).
2. Записывать введенную (в элемент управления RichtextBox) информацию в текстовый файл построчно и посимвольно.

Методика выполнения задания

1. Необходимо разработать главную форму, с помощью которой можно выбрать режим чтения или записи.
2. Для каждого из режимов работы с приложением (чтение или запись) разработайте форму для возможности выбора построчного или посимвольного чтения/записи.

На рис. 8.1 приведен пример такой формы.

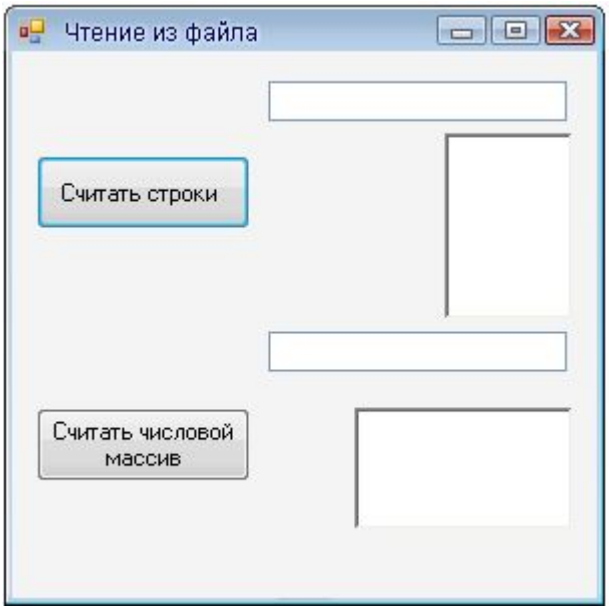

Рис.8. 1

3. Подключение к пространству имен следует выполнить сразу после объявления класса - Public Class Form1:

```
Imports System
Imports System.IO
```

4. Создайте в каталоге Bin\Debug вашего проекта текстовый файл с именем File и введите туда построчно информацию.
5. Код (метод) на событие «Нажатие кнопки **Считать строки**»

```
  Try
        ‘ Create an instance of StreamReader to read from a file.
        Dim sr As StreamReader = New StreamReader("File.txt")
        Dim line As String
        ' Read and display the lines from the file until the end of the file is reached.
        Do
           line = sr.ReadLine()
           RichTextBox1.Text = RichTextBox1.Text & line & Chr(13)
        Loop Until line Is Nothing
        sr.Close()
     Finally
        ' Let the user know that all is OK.
        TextBox1.Text = "The file is read:"
     End Try
```

На рисунке 8.2 приведено вид окна, полученный при нажатии кнопки **Считать строки**.

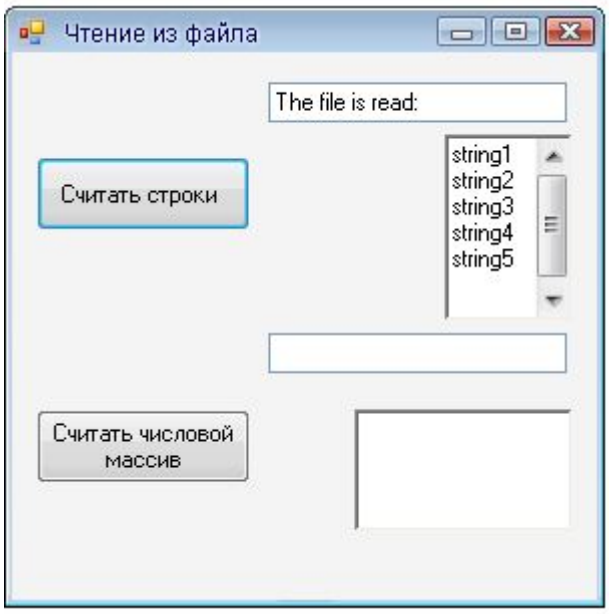

Рис.8.2

Лабораторная работа №9
Работа с процедурами Function и Sub

Цель: Научиться создавать пользовательские процедуры в VB.Net.

Задание 9.1

Используя процедуру Function, разработать приложение для вычисления площади прямоугольника.

Задание 9.2

Используя процедуру Sub, разработать приложение, которое для введенной суммы в центах, показывает, сколько долларов и сколько центов в сумме.

Методика выполнения задания 9.1

1. Спроектировать форму (см. Рис.9.1)

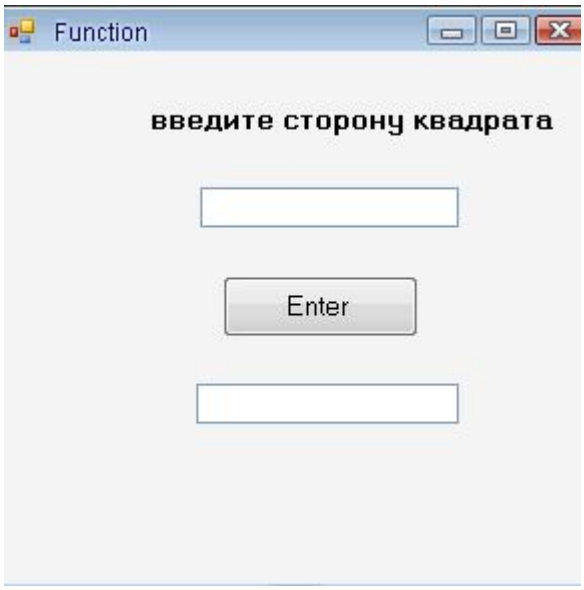

Рис.9.1 Главная форма приложения

2. Добавить в класс Формы процедуру Function (метод):

```
Private Function square(ByVal a As Integer) As Integer
        square = a ^ 2
End Function
```

3. Написать на событие Button1_Click (кнопка Enter) следующий код:

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As_ System.EventArgs) Handles Button1.Click
        TextBox2.Text = square(CInt(TextBox1.Text))
End Sub
```

Методика выполнения задания 9.2

1. Спроектировать форму (см. Рис.9.2)
2. Добавить в класс Формы процедуру Sub (метод):

```
Private Sub DollarsAndCents(ByVal totalCents As Integer, ByRef Dollars As Integer,_
                            ByRef CentsLeft As Integer)
        Dollars=totalCents\100
        CentsLeft=totalCents Mod 100
End Sub
```

3. Написать на событие Button1_Click (кнопка Calculate) следующий код:

```
Dim originalCents, wholeDollars, centsLeft As Integer
originalCents=Cint(TextBox1.text)
DollarsAndCents(originalCents,  wholeDollars, centsLeft )
Label1.text=Cstr(wholeDollars)
Label2.text=Cstr(centsLeft)
```

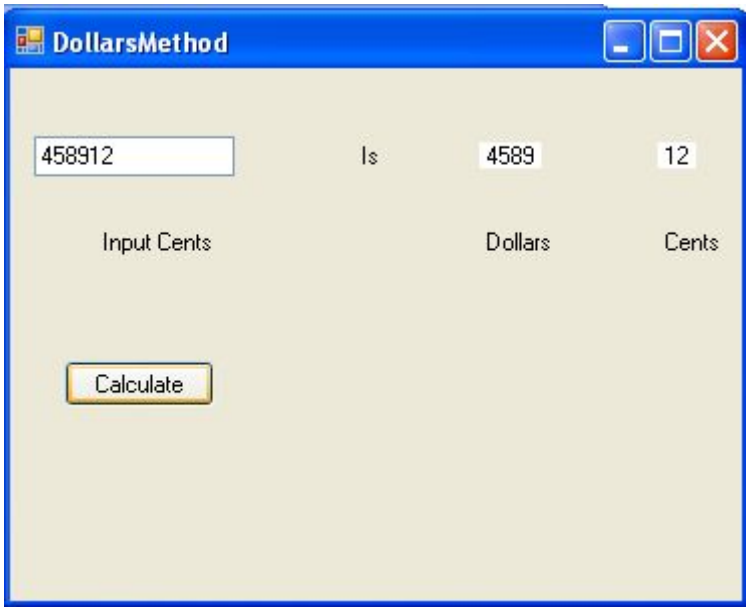

Рис.9.2

Задание для самостоятельной работы

Создайте приложение для обменного пункта валюты. На вход подается сумма в иностранной валюте (доллар, евро, казахский тенге, узбекский сум) и курс по отношению к сому. Приложение переводит сумму в сомы. Перевод оформить в виде Sub процедуры.

Лабораторная работа №10
Объектно-ориентированное программирование в VB.Net

Цель: Научиться создавать классы в VB.Net и работать с ними.

Краткие теоретические сведения

Каждый класс содержит набор полей, методов, свойств, событий (обобщенно их называют членами класса). Рассмотрим кратко каждую из этих составляющих класса.

- Поля — переменные, принадлежащие классу или экземпляру класса. Принадлежность к классу или экземпляру класса характерна не только для полей, но и для методов, событий и свойств.
- Методы — процедуры и функции класса.
- Свойства — синтаксическая надстройка, позволяющая осуществлять в форме вызов функции, аналогичной чтению/записи переменной. Например, можно объявить свойство Возраст, и при попытке записи в него отрицательного значения выдавать ошибку. На самом деле это не чисто синтаксическая надстройка.
- События—синтаксическая надстройка, поддерживаемая компилятором и средой Visual Basic 2008, которая позволяет вызывать методы других объектов, подписавшихся на данное событие. Например, подписавшись на событие Нажатие объекта Кнопка, подписавшийся объект каждый раз при нажатии кнопки будет получать уведомление (в виде вызова метода).

Каждая из этих составляющих класса, а также сам класс могут иметь так называемые модификаторы доступа, которые указывают область ее видимости. Значения модификаторов могут быть следующими:

- Public — открытый класс или член класса. Доступ к нему разрешен из любого места кода;
- Private — класс или член класса, доступный только из контекста, в котором он объявлен, и во всех вложенных контекстах. Это значит, что если, например, свойство объявлено с модификатором Private, то оно доступно только из того же самого класса и из вложенных в него классов;
- Friend — класс или член класса, доступный только внутри той же сборки, в которой объявлен. Сборка — полностью самостоятельная единица приложения .NET. В Visual Basic 2008 сборка обычно соответствует всей программе, поэтому данный модификатор можно воспринимать как указание видимости только в пределах программы;
- Protected — член, доступный только из самого класса и из наследующих классов. Данный модификатор применим только к членам классов;
- Protected Friend — объединение областей видимости Protected и Friend. Член доступен в той же сборке или в наследующих классах.

Наряду с модификаторами доступа, регламентирующими видимость, члены класса могут содержать модификаторы, устанавливающие их принадлежность к классу или к экземпляру класса. Члены класса, принадлежащие классу, называют разделяемыми (shared) или статическими (static) членами. Члены, которые принадлежат экземпляру, называются экземплярными (instance). Рассмотрим следующий пример:

```
Module MyModule
Public Class Human
Private humanAge As Integer
Private humanName As String
Private humanNick As String
' Возраст
```

```
Public Property Age() As Integer
Get
Return humanAge
End Get
Set(ByVal Value As Integer)
If Value < 0 Or Value > 200 Then
   Throw New ArgumentException("Age must be between 0" + " and 200 years")
End If
humanAge = Value
End Set
End Property
'Имя
Default Public Property Name(ByVal NickName As Boolean) As String
Get
    If NickName Then
        Return humanNick
    Else
        Return humanName
    End If
End Get
Set(ByVal Value As String)
    If NickName = True Then
        humanNick = Value
    Else
        humanName = Value
    End If
End Set
End Property
End Class
' Точка входа в программу
Sub Main()
Dim a As New Human()
a.Name(True) = "some name"
a(True) = "some name" ' Эта строка эквивалентна предыдущей
Console.WriteLine(a.Name(True))
a.Age = 5
Console.WriteLine(a.Age)
a.Age = -2
End Sub
End Module
```

В этом примере создается класс, содержащий данные о человеке. В частности, класс содержит данные о возрасте и имени человека, причем имя может быть двух типов — обычное имя и дополнительное (например, имя учетной записи почтового клиента). Эти данные спрятаны от непосредственного доступа (объявлены с модификатором Private) — доступ открыт с помощью свойств. Свой-

ство Age позволяет установить/считать возраст человека, при этом проводится проверка на допустимость введенного значения. Такую проверку было бы невозможно сделать, если бы
возраст был просто открытой переменной.
Также в классе свойство Name объявлено, как свойство по умолчанию. Это позволяет обращаться к данному свойству, не указывая его имени.

В приведенном примере оба свойства позволяют выполнять чтение и запись переменных. Однако можно сделать так, чтобы свойство было доступно только для чтения или только для записи. Для этого достаточно опустить один из блоков Get или Set, указав при этом слова Readonly или WriteOnly (Readonly, если есть только блок Get, и WriteOnly, если есть только блок Set). Студенту предоставляется возможность сделать это самостоятельно.

Индивидуальные задания для самостоятельной работы

По завершении курса обучения по дисциплине **"Средства визуальной разработки приложений "** каждый студент должен выполнить индивидуальное задание, которое выбирает он сам по согласованию с преподавателем. При выполнении задания должны быть применены знания по технологии работы с текстовыми файлами (чтение и запись данных), построению графиков, работе с базами данных. Перечень индивидуальных заданий приведен ниже.

1. Создать переводчик для нескольких языков. Он должен предусматривать выбор направления перевода (как в программе Prompt). Исходные слова должны выбираться из выпадающего списка. Результат должен приводиться в текстовом поле. В окне переводчика должны быть предусмотрены кнопки для быстрого вызова команд.
2. Создать тест на знание английского языка. Переводимое слово можно выбирать из выпадающего списка. Должно быть предусмотрено использование картинок. Вариант перевода можно также выбирать из выпадающего списка.
3. Создать программу – помощник для решения квадратного и линейного уравнения с одной переменной. На вход подаются коэффициенты полинома. Выводятся дискриминант и корни.
4. Создать программу – помощник для расчета площади геометрических фигур: квадрата, прямоугольника, параллелограмма, ромба.
5. Написать программу, которая производит сортировку числовых данных с выбором направления сортировку. Использовать встроенную процедуру VB и какой - либо алгоритм сортировки. Данные должны вводиться в тестовое поле.
6. Написать программу, которая производит поиск заданного числа. Использовать вызов готовой процедуры VB и запрограммировать алгоритм бинарного поиска. Данные должны вводиться в тестовое поле.
7. Написать программу, которая позволяет вывести возраст пользователя по введенному дню, месяцу и году рождения. Месяц рождения вводить, используя выпадающий список.
8. Write a program which will input the user’s name and a message, display the two items concatenated in a label, and change the format of the label. Using option buttons and check boxes for selection, the user can make the label bold, underlined, or

italic, and change its color. Include command buttons to display message in the label, clear text boxes and label, and exit. The sample of the main form is on the picture 10 (рис.10).

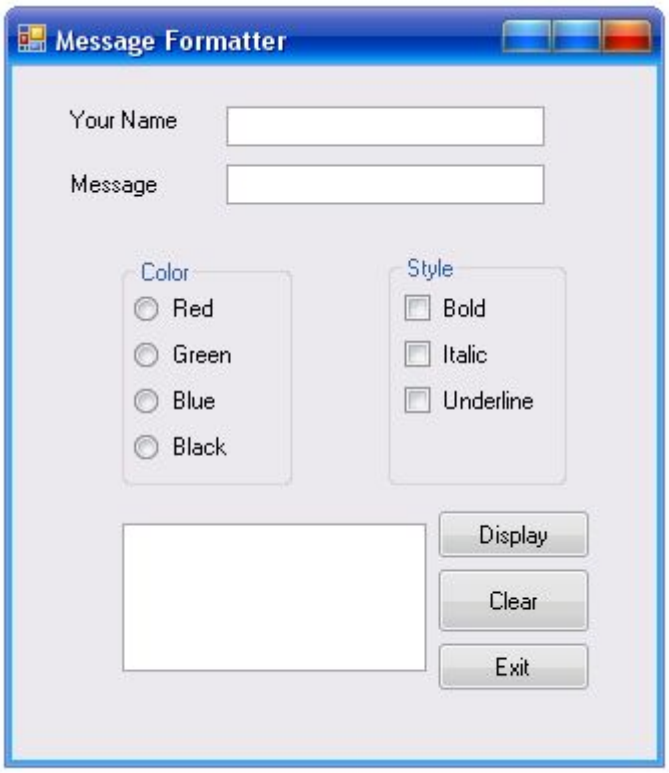

Рис.10

9. Calculate the amount due for an order. For an order, the user should enter the following information into text boxes: customer name, address, city, state and ZIP code. An order may consist of multiple items. Foe each item, the user will enter the product description, quantity, weight and price into text boxes.

You will need command buttons for Next Item, Update Summary and Exit. For the Next Item button, validate the quantity, weight, and price. Each must be present and numeric. For any bad data, display a message box. Calculate the charge for the current item and add the charge and weight into the appropriate totals. Do not calculate shipping and handling on individual items; rather, calculate shipping and handling on the entire order.

When the Update Summary button is clicked, calculate the sales tax, shipping and handling, and the total amount due for the order. Sales tax is 8% of the total charge and is charged only for shipments to a California address. Do not charge sales tax on the shipping and handling charges.

The shipping and handling charges depend on the weight of the products. Calculate the shipping charge as $.25 per pound and add that amount to the handling charge (taken from the following table).

Weight	Handling
Less than 10 pounds	$1.00
10 to 100 pounds	$3.00
Over 100 pounds	$5.00

10.Display the entire amount of the bill in labels titled ***Dollar amount due***, ***Sales tax, Shipping and handling***, and ***Total amount due***.

11.Calculate prices and discounts for book sold. The company is currently having a big sale, offering discount on all books (different percent on different book) In the project calculate the amount due for a quantity of books, determine the discount, and deduct the discount, giving the new amount- discount amount, the total discount, and the average discount per sale. See picture 11 (рис.11).

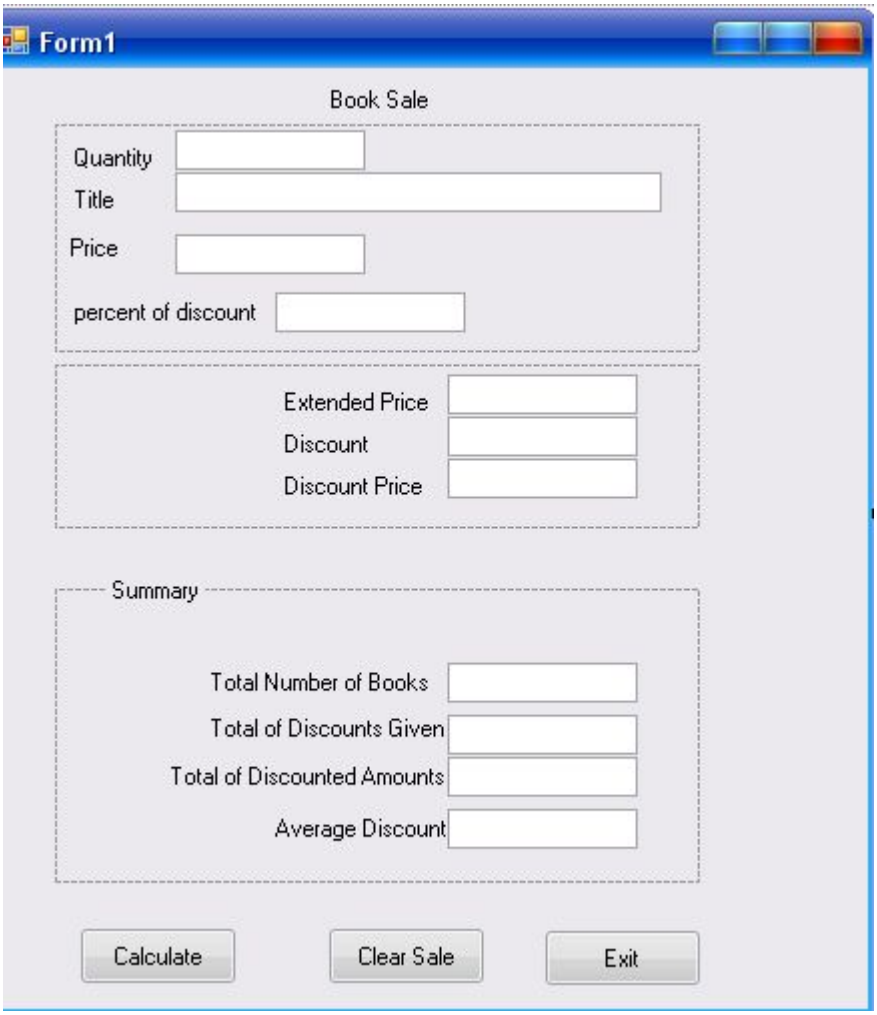

Рис.11

12. Create a project that calculates the amount due for individual orders and maintains accumulated totals for summary. Have a check box for takeout items, which are taxable (8 percent); all other orders are nontaxable. Include option buttons for 5 coffee selections. Input the prices for each. See pic.12 (рис.12).

Рис.12

13.Создайте программу – калькулятор для официанта кафе «Мороженное» по заказанным видам мороженного. Интерфейс примерно такой же, как в задании 10. Надо предусмотреть ввод возможность комбинировании различных сортов мороженного.

14.Создайте программу – калькулятор для официанта ресторана по заказанным блюдам клиента. Интерфейс примерно такой же, как в задании 10.

15.Create a project that will be used to determine the total amount due for the purchase of a vehicle. You will text boxes for the base price and the trade – in allowance. Check boxes will indicate if the buyer wants additional accessories: stereo system, leather interior, and/or computer navigation. A frame for the exterior finish will contain option buttons for Standard, Pearl zed, or Customized Detailing. See pic. 13 (Рис.13)

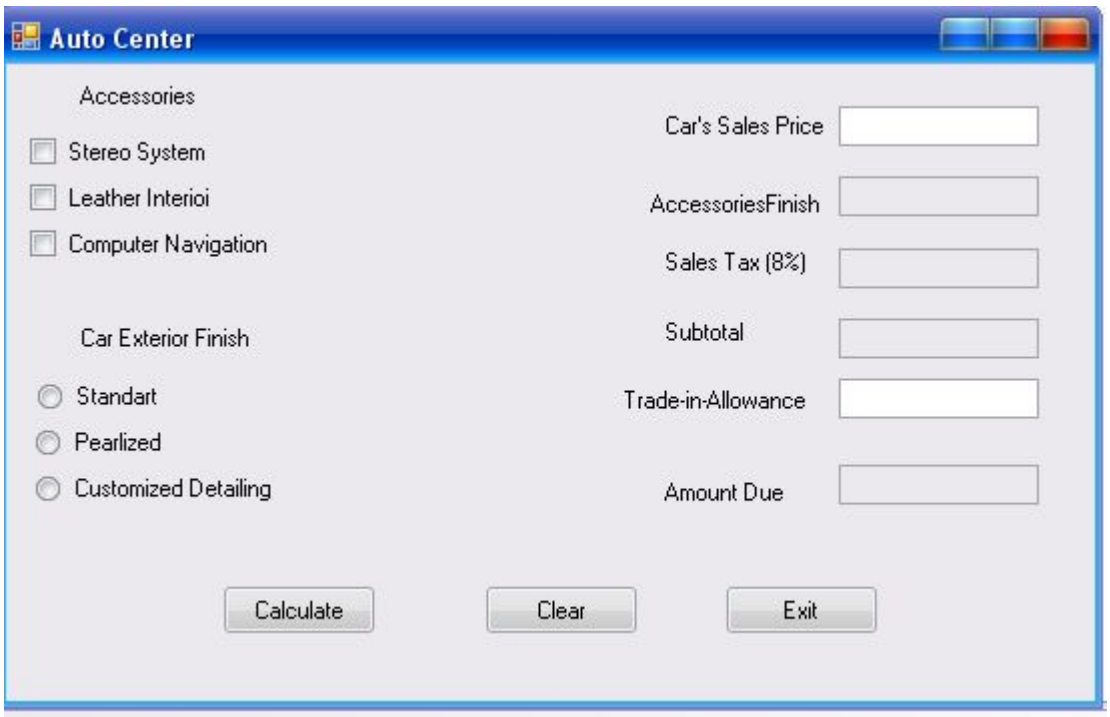

Рис.13

Литература

1. Петров В.Н. Информационные системы. – СПб.: БХВ – Петербург, 2008
2. Шевякова Д., Степанов А., Дукин А. Visual Basic 2008. – СПб.: БХВ – Петербург, 2002.
3. Маргулев А.И. Программирование на Visual Basic 5. biblioteka.net.ru
4. «Учебные материалы по Visual Basic », Biblioteka.Net.Ru
5. Duglas Bell & Mike Parr. Visual Basic.Net for students. Addison – Wesley, 2003.