Лист заданий имеет вид:

Лист заданий

1 Построить приложение оболочку ObPro_Shell с меню вида:

```
File     Collection              ADO NET                          Help
  Exit     CollectionBase          WindowsApplication               About
             Binary                  DataGrid                       Contents
             Soap                    TextBox
             XML                     Relation
           CollectionArrayList       Edit
             ArrayList
           CollectionHashTable
             HashTable
```

2 Построить проекты в среде Visual Basic .NET

Binary

Soap

Нет XML

DataGrid

Нет TextBox

Нет Relation

Нет Edit

3 Условия задач: получить у преподавателя.

4 Построить пояснительную записку к проекту курсовой работы
Report_KursRabota_ObjPro_Ek_02_b_Sidorov.doc

5 Представить архивы:

KursRabota_ObPro_Ek_02_b_Sidorov_Shell.rar

KursRabota_ObPro_Ek_02_b_Sidorov_Binary.rar

KursRabota_ObPro_Ek_02_b_Sidorov_Soap.rar

Нет KursRabota_ObPro_Ek_02_b_Sidorov_XML.rar

KursRabota_ObPro_Ek_02_b_Sidorov_DataGrid.rar

Нет KursRabota_ObPro_Ek_02_b_Sidorov_TextEdit.rar

Нет KursRabota_ObPro_Ek_02_b_Sidorov_Relation.rar

Нет KursRabota_ObPro_Ek_02_b_Sidorov_Edir.rar

KursRabota_ObPro_Ek_02_b_Sidorov_Report.rar

Задание выдано 1.09.2004г.

Студент гр. Эк 02 б - 3з Петров А.В.

Реферат работы может имеет вид:

РЕФЕРАТ

Стр. 15 Рис. 5

Целью работы является освоение объектно-ориентированных средств создания программных моделей экономических систем.

Объектом исследования являются объектно-ориентированных средств создания программных моделей экономических систем.

Полученные результаты могут быть использованы при построении программных моделей экономических систем, что является в настоящее время ведущей технологией создания программных моделей экономических систем.

Полученные навыки могут быть использованы при построении программных моделей экономических систем.

НАБОР, СЕРИАЛИЗАЦИЯ, ИСТОЧНИК ДАННЫХ, НАБОР ДАННЫХ, ПРОВАЙДЕР ДАННЫХ

Оглавление работы имеет вид:

ОГЛАВЛЕНИЕ
ВВЕДЕНИЕ
1 Наборы данных

1.1 Задача «Binary»

1.2 Задача «SOAP»

1.3 Задача «ArrayList»

1.4 Задача «HashTable»

2 Базы данных

2.1 Задача «DataGrid»

ЗАКЛЮЧЕНИЕ
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1

ВВЕДЕНИЕ

Целью выполнения курсовой работы по дисциплине «Объектное программирование» является:

закрепление навыков работы в Windows Development Environment;

закрепление навыков отладки приложений;

закрепление навыков организации контекстной помощи приложений;

освоение расширенных возможностей организации конструкторов классов;

освоение расширенных возможностей организации свойств классов;

освоение расширенных возможностей организации методов классов;

освоение расширенных возможностей организации механизма наследования классов;

освоение механизма организации абстрактных классов;

Приобретенные навыки объектно-ориентированного программирования могут быть использованы при создании программных моделей экономических систем.

1 НЕОБХОДИМЫЕ СВЕДЕНИЯ

1.1 Наборы данных

1.1.1 Пример «CollectionBaseBinary»

Условие

Дано вербальное (словесное) описание основного и альтернативных сценариев (проблемных ситуаций) прецедента (варианта использования) некоторой ЭС.

Задание 1
Построить абстракцию проблемы и ее решения, представив их отдельным классом, со свойствами Q (Question) и A (Answer), String типа.

Свойство Q – название проблемы.

Свойство A – решение проблемы.

Задание 2.1
Построить класс-набор Problems на основе стандартного класса CollectionBase, пометив в набор объекты - проблемные ситуации системы и выполнить сериализацию набора Problems в формате Binary в файле QABinary.xxx

Задание 2.2
Построить класс-набор Problems на основе стандартного класса CollectionBase, пометив в набор объекты - проблемные ситуации системы и выполнить сериализацию набора Problems в формате XML в файле QAXML.xxx

Задание 2.3
Построить класс-набор Problems на основе стандартного класса CollectionBase, пометив в набор объекты - проблемные ситуации системы и выполнить сериализацию набора Problems в формате SOAP в файле QASOAP.xxx

Задание 2.1a
Отобразить исходный код файла QABinary.xxx

Задание 2.2a
Отобразить исходный код файла QAXML.xxx

Задание 2.3a
Отобразить исходный код файла QASOAP.xxx

Указания к решению

Вербальное описание прецедента

Юридическое лицо – малое частное предприятие, работающее по упрощенной системе налогообложения со ставкой 10%, осуществляет коммерческую деятельность.

Арендует помещение, оборудование.

Имеет договор с Энергонадзором на пользование электроэнергией.

Основной поток событий варианта использования «Ежемесячный отчет в налоговой службе»:

1 Определение списка проверки задолженностей - «Бегунка»

2 Проверка «Бегунка» на долг по «Оплате единого налога»

3 Проверка «Бегунка» на долг по «Оплате подоходного налога»

4 Проверка «Бегунка» на долг по «Погашению штрафных санкций»

5 Изготовление формы 8ДР

6 Проверка Информационной карты

7 Проверка формы 8ДР

8 Проверка формы 1ПП

9 Проверка формы Расчет Подоходного налога

1 Альтернативный поток событий «Есть долг по «Оплате единого налога»

1 Определение списка проверки задолженностей - «Бегунка»

2 Проверка «Бегунка» на долг по «Оплате единого налога»

3 Устранение долга по «Оплате единого налога»

4 Проверка «Бегунка» на долг по «Оплате подоходного налога»

5 Изготовление формы 8ДР

6 Проверка Информационной карты

7 Проверка формы 8ДР

8 Проверка формы 1ПП

9 Проверка формы Расчет Подоходного налога

Альтернативный поток событий «Есть долг по «Оплате подоходного налога»

1 Определение списка проверки задолженностей - «Бегунка»

2 Проверка «Бегунка» на долг по «Оплате единого налога»

3 Проверка «Бегунка» на долг по «Оплате подоходного налога»

4 Устранение долга по «Оплате подоходного налога»

5 Проверка «Бегунка» на долг по «Погашению штрафных санкций»

6 Изготовление формы 8ДР

7 Проверка Информационной карты

8 Проверка формы 8ДР

9 Проверка формы 1ПП

10 Проверка формы Расчет Подоходного налога

Альтернативный поток событий «Есть долг по Расчету ПодохНалога»

1 Определение списка проверки задолженностей - «Бегунка»

2 Проверка «Бегунка» на долг по «Оплате единого налога»

3 Проверка «Бегунка» на долг по «Оплате подоходного налога»

4 Проверка «Бегунка» на долг по «Погашению штрафных санкций»

5 Изготовление формы 8ДР

6 Проверка Информационной карты

7 Проверка формы 8ДР

8 Проверка формы 1ПП

9 Проверка формы «Расчет Подоходного налога»

10 Устранение долга по «Расчет Подоходного налога»

Указание

Программный код проекта разместить на одном листе модуля, назвав его Module_Binary.

Указание

Принять следующий сценарий тестирования:

1 Создать объекты типа Problem
2 Определить свойства Name и Answer.

3 Отобразить значения свойств
4 Отобразить число объектов набора до и после их помещения в набор
5 Итерируя по элементам набора, отобразить значение свойства Name
6 Выполнить сериализацию
7 Удалить второй элемент из набора
8 Итерируя по элементам набора, отобразить значение свойства Name
9 Удалить все объекты из набора
10 Отобразить число объектов набора
11 Выполнить десериализацию набора
12 Итерируя по элементам набора, отобразить значение свойства Name

Указание

В качестве процедуры сериализации принять следующий код:

```
Sub SerializeToBinary(ByVal anCollection As Problems, ByVal fName As String)
  Dim fStream As FileStream
  Dim myBinaryFormatter As New Formatters.Binary.BinaryFormatter()
  Try
      fStream = New FileStream(fName, FileMode.Create, FileAccess.Write)
      myBinaryFormatter.Serialize(fStream, anCollection)
  Catch e As Exception
      Throw e
  Finally
      If Not (fStream Is Nothing) Then fStream.Close()
  End Try
End Sub
```

Указание

В качестве процедуры десериализации принять следующий код:

```
Function DeSerializeFromBinary(ByVal fName As String) As Problems
  Dim fStream As New FileStream(fName, FileMode.Open, FileAccess.Read)
  Dim myBinaryFormatter As New Formatters.Binary.BinaryFormatter()
  Try
    Return CType(myBinaryFormatter.Deserialize(fStream), Problems)
  Catch e As Exception
    Throw e
  Finally
    If Not (fStream Is Nothing) Then fStream.Close()
  End Try
End Function
```

Указание

До кода модуля импортировать следующие пространства имен:

```
Imports System.IO
Imports System.Collections
Imports System.Runtime.Serialization
Imports System.Runtime.Serialization.Formatters
```

1.1.2 Пример «CollectionBaseSOAP»

Условие

Условие задачи тоже, что и в задаче «CollectionBaseBinary».

Указания к решению

Указание

В задаче CollectionBaseSOAP требуется подключить пространство имен System.Runtime.Serialization.Formatters.Soap для этого надо выполнить Мн_Project - Км_Add Reference… - Сп_Component Name – ДвЩлч_System.Runtime.Serialization.Formatters.Soap – Кн_Ок

Указание

Строку Dim myBinaryFormatter As New Formatters.Binary.BinaryFormatter() заменить на строку

Dim mySoapFormatter As New Formatters.Soap.SoapFormatter()

Указание

Строку myBinaryFormatter.Serialize(fStream, anCollection)заменить на строку

mySoapFormatter.Serialize(fStream, anCollection)

1.1.3 Пример «CollectionArrayList»

Условие

Дан стандартный класс `ArrayList. Требуется:`

```
- объявить набор
- заполнить набор строками
- отобразить число элементов набора
- отобразить элементы набора
- добавить еще один элемент в набор на третье место
- отобразить число элементов набора
- отобразить элементы набора
- отсортировать элементы набора
- копировать набор в массив
- отобразить массив
- очистить набор
- отобразить число элементов набора
```

Указания к решению

Указание 1

```
Объявление набора
  Dim arlA As New ArrayList()
```

Указание 2

```
Заполнение набора строками
  For i = 0 To n - 1
    arlA.Add(InputBox("a(i) = ", "", "qqq"))
  Next i
```

Указание 3

```
Отображение числа элементов набора
  MsgBox("count 1 = " & arlA.Count)
```

Указание 4

```
Отображение элементов набора
  For i = 0 To n - 1
    MsgBox("arlA.Item(i) = " & arlA.Item(i))
  Next i
```

Указание 5

```
Добавление элемента в набор
  arlA.Insert(2, InputBox("a(i) = ", "", "PPP"))
```

Указание 6

```
Сортировка набора
  arlA.Sort()
```

Указание 7

```
Копирование набора в массив
  arlA.CopyTo(a)
```

Указание 8

```
Очищение набора
  arlA.Clear()
```

1.1.4 Пример «CollectionHashTable»

Условие

Дан стандартный класс ArrayList. Требуется:
- объявить набор
- заполнить набор ключами равными номеру элемента и значениями - строками
- отобразить число элементов набора
- отобразить элементы набора
- добавить еще один элемент в набор
- отобразить число элементов набора
- отобразить элементы набора оператором For-Next
- отобразить элементы набора, используя интерфейс IDictionaryEnumerator и метод GetEnumerator. Итерации по набору выполнить оператором While - End While
- удалить элемента набора с заданным ключом
- отобразить элементы набора
- копировать набор в массив
- отобразить массив
- очистить набор
- отобразить число элементов набора

Указания к решению

Указание 1
Заполнение набора ключами равными номеру элемента и значениями - строками
```
  For i = 0 To n - 1
      htA.Add(i, InputBox("a(i) = ", "", "qqq"))
  Next i
```
Указание 2
Добавляение элемента в набор
```
  htA.Add(12, InputBox("a(i) = ", "", "PPP"))
```
Указание 3
Отображение набора, используя интерфейс IDictionaryEnumerator и метод
```
   GetEnumerator, который возвращает IDictionaryEnumerator, который, в свою очередь,
   позволяет итерировать по наботу Hashtable.
  Dim myEnumerator As IDictionaryEnumerator = htA.GetEnumerator()
```
Указание 4
Итерации по набору, используя While - end While
```
  While myEnumerator.MoveNext()
    MsgBox("Имя ключа - " & myEnumerator.Key & "   Объект - " & myEnumerator.Value)
  End While
```
Указание 5
Удаление элемента набора с ключом равным "qqq"
```
  htA.Remove("qqq")
```
Указание 6
Копирование набора в массив
```
  Dim myArray As Array = Array.CreateInstance(GetType(String), 15)
  htA.Values.CopyTo(myArray, 0)
```
Указание 7
Очистка набора
```
  htA.Clear()
```

Оператор While – End While

Выполнение тела цикла пока условие истинно

ОбщВид

```
While Условие
   [Тело цикла]
End While
```

Где

Условие
Выражение. Изменяется от Истина до Ложь. Значение выражения равное Nothing понимается как Ложь

[Тело цикла]
Не обязательно.

Работа

- выполняется Тело цикла

- проверяется Условие

You can nest While loops by placing one loop within another.

Пример

```
Dim Counter As Integer = 0
While Counter < 20
  Counter += 1
End While
```

2.1 Базы данных

2.1.1 Пример «DataGrid»

Условие
Дана база данных Tovar.mdb

Задание 1
Построить базу данных Tovar.mdb средствами MS Access, содержащую одну таблицу «Товары» с полями: Номер Наименование Цена Количество Адрес

Задание 2
Создать соединение с источником данных - базой данных Tovar.mdb

Задание 3
Создать провайдер Адаптер данных на основе OleDbDataAdapter

Задание 4
Создать Набор данных на основе DataSet

Задание 5
Заполнить Набор данных таблицей «Товары»

Задание 6
Отобразить Набор данных в элементе DataGrid

Указания к решению

Указание 1
За кнопкой Close закрепить код: Me.Close()

Указание 2
Свойство ConnectionString объекта OleDbConnection определить так:

```
Dim myConnectionString As String = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Устройств\Путь\Tovar.mdb"

Dim myConnection As New OleDbConnection(myConnectionString)
```

Указание 3
Адаптер данных можно определить. Например, так:

```
Dim myDataAdapter As New System.Data.OleDb.OleDbDataAdapter()
Dim myCommand As New System.Data.OleDb.OleDbCommand()
    myCommand.Connection = myConnection
    myCommand.CommandText = "SELECT * FROM Товары"
    myDataAdapter.SelectCommand = myCommand
Dim myDataSet As New DataSet()
```

Указание 4
Соединение открыть методом Open()

```
myConnection.Open()
```

Указание 5
Набор данных можно определить, например, так:

```
Dim myDataSet As New DataSet()
```

Указание 6
Заполнение Набора данных

```
myDataAdapter.Fill(myDataSet, "Товары")
```

Указание 7
После заполнения Набора данных его можно отобразить

```
MsgBox("Имя таблицы БД - " & myDataSet.Tables(0).TableName)
MsgBox("Число строк таблицы БД - " & myDataSet.Tables("Товары").Rows.Count)
MsgBox("Число столбцов таблицы БД - " & myDataSet.Tables("Товары").Columns.Count)
```

Указание 8
Отобразить Набор данных в элементе DataGrid можно так:

```
myDataGrid.DataSource = myDataSet.Tables("Товары")
```
Указание 9
```
Добавление несуществующих столбцов
'Обычный столбец "Цена"
myDataSet.Tables("Товары").Columns.Add("Cost").Caption = "ЦЕНА"
myDataGrid.DataSource = myDataSet.Tables("Товары")
MsgBox("Добавлен несуществующий  обычный столбец Цена         " &
myDataSet.Tables("Товары").Columns(6).ColumnName)
'Обычный столбец "Количество"
myDataSet.Tables("Товары").Columns.Add("Count").Caption = "КОДИЧЕСТВО"
myDataGrid.DataSource = myDataSet.Tables("Товары")
MsgBox("Добавлен несуществующий  обычный столбец Количество   " &
myDataSet.Tables("Товары").Columns(7).ColumnName)
'Обычный столбец "Адрес"
myDataSet.Tables("Товары").Columns.Add("АДРЕС").Caption = "адрес"
myDataGrid.DataSource = myDataSet.Tables("Товары")
MsgBox("Добавлен несуществующий  обычный столбец Адрес        " &
myDataSet.Tables("Товары").Columns(8).ColumnName)
'Вычисляемый столбец "КолЦена"
myDataSet.Tables("Товары").Columns.Add("КолЦена").Caption = "КолЦена"
myDataGrid.DataSource = myDataSet.Tables("Товары")
myDataSet.Tables("Товары").Columns(9).Expression = "Count*Cost"
MsgBox("Добавление несуществующего вычисляемый столбец КолЦена" &
myDataSet.Tables("Товары").Columns(9).ColumnName)
'Автовозрастающий столбец "ID"
myDataSet.Tables("Товары").Columns.Add("ID").Caption = "ID"
'myDataSet.Tables("Товары").Columns("ID").AutoIncrement = True
'myDataSet.Tables("Товары").Columns("ID").AutoIncrementSeed = 1
'myDataSet.Tables("Товары").Columns("ID").AutoIncrementStep = 1
myDataGrid.DataSource = myDataSet.Tables("Товары")
MsgBox("Добавлен несуществующий автовозрастающий столбец ID   " &
myDataSet.Tables("Товары").Columns(10).ColumnName)

'Определение первичного ключа
'Предполагаем, что более трех полей как ключ использовать не будем
Dim myPrimaryKeyArray(3) As DataColumn
myPrimaryKeyArray(0) = myDataSet.Tables("Товары").Columns("ID")
'myPrimaryKeyArray(1) = myDataSet.Tables("Товары").Columns("...")
'myPrimaryKeyArray(2) = ...
MsgBox("Число стобцов = " & myDataSet.Tables("Товары").Columns.Count)
```
Указание 10

В конце работы соединение надо закрыть

```
myConnection.Close()
```

и освободить ресурсы соединения myConnection.Dispose()

Заключение

Рассмотренные средства системы VB .NET достаточно эффективно позволяют создавать виртуальные модели реальных систем.

В результате тестирования программных моделей фрагментов ЭС выявлены особенности работы инструментальных средств.

В работе построены виртуальные модели фрагментов ЭС, которые могут быть использованы при построении ПМ ОМ ЭС.

Список использованных источников

1 Корнелл Г., Моррисон Дж. Программирование на VB.NET: учебный курс.– СПб.: Питер, 2002.–400 с.:

2 Рохилла С., Натан С., Мелхотра С. Microsoft ADO NET: разработка профессиональных проектов. – СПб.: БХВ – Петербург, 2003. – 768с.: ил.