Noorul Islam College of Arts and Science, Kumarcoil
Accredited with 'B' Grade by NAAC
(Affiliated to Manonmaniam Sundaranar University, Tirunelveli)
Thuckalay, Kanyakumari Dist

Dot NET Technologies
Sub.Code:SMCS53

Subject Handled by
Dr.R.Rajalakshmi
Asst.Prof & Head,Dept of Computer Science
Noorul Islam College of Arts and Science

Dr. S.Perumal
Principal, Noorul Islam College of Arts and Science, Kumaracoil.

UNIT-III

Using the .NET Framework Class Library

- **Data Collection**
- **File I/O**
- **Event logging**
- **Message queuing**
- **Text Handling**
- **Internet communication**
- **XML data handling**
- **Internet e-mail**

Using Data Collection
(System Collection)

- **Arrays: Ordered or unordered collection of objects,**
  first element has the index 0.
  Random access in the array is allowed.

Stacks: collection of elements,
insertion and deletion on the top(ie only one end,
LIFO structure, eg.,recursion rely on stack operation

- **Queues: queues are structures,**
  allows for insertion of elements of one-end and removal of elements from the other. eg., customers waiting for service at a bank window.

Hash table: a set of elements which have a unique Key value rather than an ordinal index.
eg., it follows a fastlookups of information without linearly searching.

The ArrayList class

- System.Collection.ArrayList class
- Program creates two arrays
- Two arrays of integers
- Merges the two sorts the merge list
- Search element from a list

```
Imports system
Imports system.collections
Imports Microsoft.VisualBasic
Module module1
 Sub Main()
   Dim alist1 as Arraylist
   Dim alist2 as Arraylist
   Dim intsearchchoice As Integer
   Dim intlocalindex As Integer
 ’ create the new list
  alist1=CreateArraylist()
  alist2=CreateArraylist()
 ’original two Arralists
printArraylist(alist1)
printArraylist(alist2)
 ’show combined Arraylist
alist1.AdRange(alist2)
PrintArraylist(alist1)
’sort the list
Alist1.sort()
printArraylist(alist1)
‘ search the list
Console.Write(“enter a  element to search
    for:”)
Intsearchchoice=CInt(console.ReadLine())
Intlocalindex=_alist1.BinarySearch()),_Int
    searchchoice,Nothing)
If Intlocalindex<>-1 then
Console.Write(“match found at
    index:{0}”),_ Intlocalindex
Else
Console.Write(“no match found,”)
End If
End sub
```

```
Sub printArraylist(ByVal arylist as Arraylist)
      Dim intElement As integer
      For Each intElement In Arylist
       Console.Write(intElement.ToString()&” “)
       Next
     Console.WriteLine()
End sub
 Function CreateArrayList() As Arraylist
      Dim aNewArraylist As New Arraylist()
      Dim intIndex As Integer
      Dim rndRandom As New Radom()
     Randomize()
 For IntIndex=0 to 9
     aNewArraylist.Add(rndRandom.Next(0,99)
Next
   CreateArrayList= aNewArraylist
End Function
End Module.
```

The Stack Class

- System.Collection.Stack implements the stack data structure
- Undo feature that allows user to revert back to a previous state, data entry error or unintensional error
- When # is encountered pop operation
- When . Is encountered stop and exit

```
Imports  System
Imports  System . Collections
Imports Microsoft.VisualBasic
Module Module1
Sub Main()
 Dim stkUndoStack As New Stack()
 Do While (AddPhrase(stkUndoStack))
 Loop
 Console.WriteLine("Final sentence is:")
 DisplayStackContents(stkUndoStack)
End sub
Function AddPhrase(ByRef UndoStack As
    Stack) As Boolean
    Dim strWordorPhrase As String
    Dim strPhrase As String
    AddPhrase=True
    Console.Write("Add a word or phrase:")
strWordOrPhrase=Console.Readline()
select case strWordOrPhrase
    case"."
    AddPhrase=false
    Exit Function
    Case"#"
    If UndoStack.Count>0 Then
    Console.WriteLine("Undo Phrase:"&
    _UndoStack.Peek())
    strPhrase=UndoStack.pop()
    AddPhrase=True
    Else
    Exit Function
    End if
Case else
    UndoStack.Push(strWordOrPhrase)
End select
DisplayStackContents(UndoStack)
End Fuction
```

```
Sub  DisplayStackContents(ByRef  UndoStack As Stack)
        Dim strPhrase As String
        Dim aryStandardArray As object()
        Dim intIndex As Integer
        Console.Write("Current Contents :")
        aryStandardArray=UndoStack.ToArray()
For intIndex=UBound(aryStandardArray) To 0 Step-1
        Console.Write(AryStandardArray(intIndex) &"")
Next
        Console.Write(ControlChars.crLF)
End Sub
End module
```

The Queue class

- System.Collection.Queue implements a queue data structure
- Queue are used to model data that is stored in FIFO
- Eg., documents being submitted to a printer called printqueue

```
Imports System
Import System.Collections
Imports Microsoft.VisualBasic
Module Module1
Public Class DocRecord
    Public DocTitle As String
    Public DocOwner As String
    Public DateTimeSubmitted As Date
Sub New(Optional Byval DTittle=" ",
    _Optional  ByVal,Downer="")
    DocTitle=Dtitle
    DocOwner=Downer
    DateTimeSubmitted= now()
End Sub
End Class
Function Intialize Queue() As queue
    Dim objDocQueue As New Queue()
    Dim objDoc As Doc Record
    Dim intIndex As Integer
For intIndex = 1 to 5
    objDoc = New DocRecord()
with obkDoc
    .DoOwner ="Joe"
    .DocTitle ="Docu,ment #"
    intIndex.ToString()
End With
objDocQueue.Enqueue=objDocQueue
End Function
Sub AddDocToQueue(ByVal  DocQueue  As
    Queue, ByVal Doc As DocRecord)
    DotQueue.Enqueue.(Doc)
    Console.WriteLine("{0} submitted to
    Queue ",Doc.DocTitle)
End Sub
Sub PrintDoc(ByVal  DocQueue  As Queue)
    Dim objDoc As DocRecord
    objDoc= DocQueue.DeQueue()
    Console.writeline( "{0} has    printed",
    objDoc.DocTitle)
End Sub
```

```
Sub ShowDocInQueue(ByVal
            DocQueue  As Queue)
     Dim objDoc As   DocRecord
     Console.WriteLine("Print
    Queue"&ControlChars.CrLf)
For Each objDocIn
    DocQueue.ToArray()
    Console.WriteLine("{0}"&ControlC
    hars.Tab&_"{1}"&ControlChars.Tab
    &_"{2}",_objDoc.DocTitle,_objDoc.
    DocOwner, _FormatDateTime(
    _objDoc.DateTimeSubmitted))
Next
End Sub
Sub Main()
    Dim ObjPrtQueue As Queue
    objPrtQueue = InitiallizeQueue()
    ShowDocsInQueue(objPrtQueue)
    Console.WriteLine("2 documents
printed….")
printDoc(objPrtQueue)
PrintDoc(objPrtQueue)
AddDocToQueue(objprtQueue, _
New DocRecord("New
Document","Sam"))
ShowDocsInQueue(objPrtQueue)
End Sub
End Module
```

The HashTable Class

- System.Collection.Hashtable implements an associative name value pair of array of data.
- Hash table is fast and convenient way to give the programmer search and retrieval operation with resorting.
- Eg., lookup a contact in storage area
- Using lastname and ID pair search by hashtable

```
Imports  System
Imports System.Collections
Imports Microsoft.Visualbasic
Module module1
    Public class Contactentry
    Public m_strlastname as string
    Public m_strfirstname as string
    Public m_intage integer
Sub new(optional byval lname="", _
    Optional byval fname='',_
    Optional byval age=0)
    m_strlastname=lname
    m_strfirstname=faname
    m_intage=age
End sub
End class
sub addcontact(byref contactable as hashtable)
 Dim rndrandom as new random()
 Dim intidnumber as integer
 Dim objcontactentry as new contactentry()
    console.write("enter last name: ")
    objcontactentry.m_strlastname =
    console.readline()
    console.write("enter first name:")
    objcontactentry.m_strfirstname =
    console.readline()
    console.write("enter age:")
    objcontactentry.m_intage =
    cint(console.readline())
    randomize()
    intidnumber = rndrandom.next(1,1000)
do while
    contacttable.containskey(intidnumber)
    intidnumber = rndrandom.next(1, 1000)
loop
contacttable.add(intidnumber,
    objcontactentry)
console.writeline("added contact with id: "& _
   intidnumber.tostring())
End sub
```

```
sub querycontact(byref contacttable as      end sub
    hashtable)
   dim objcontact as contactentry
    console.write("Enter id number:")
    objcontact = contacttable.item(_
    cint(console.readline()))
if not objcontact is nothing then
    console.writeline(_"contact info: " &
    controlchars.crlf & _"last name: {0}" &
    controlchars.crlf & _"first name: {1}" &
    controlchars.crlf &_"age:
            {2}"&controlchars.crlf, _
            Objcontact.m_strlastname, _
            Objcontact.m_strfirstname, _
            Objcontact.m_intage.tostring())
else
console.writeline("contact not found")
end if
```

```
sub listcontacts(byref contacttable as hashtable)
dim objcontact as contactentry
for each objcontact in contacttable.values
console.writeline(“{0}” & controlchars.tab & _”{1}” & controlchars.tab & _”{2}”,_
    objcontact.m_strlastname,_objcontact.m_strfirstname,_objcontact.m_intage.tos
    tring())
next
end sub
sub main()
    dim objmycontacts as new hashtable()
    addcontact(objmycontacts)
    addcontact(objmycontacts)
    addcontact(objmycontacts)
    listcontacts(objmycontacts)
    querycontact(objmycontacts)
end sub
end module
```

Using File I/O

- File Class encapsulates various operation on single file.
- StreamReader and StreamWriter(File Classes)
- Creation, copying, deleting, moving
- File returns special classes that are used to read to and write files. Called stream classes
- Text file opeartions
- Reading Textfiles
- Writing TextFiles

Reading text files

```
Import System
Import System.IO
Module Module1
    Sub Main()
    Dim objStreamReader= File.OpenText(“test.txt”)
    Console.write(objStreamReader.ReadToEnd())
    objStreamReader.Close()
        Catch E As Exception
          Console.WriteLine(“error occurred:”&E.Message)
        End try
    End Sub
End Module
```

Writing Text Files

```
Imports system
Imports system.IO
Module module1
 Sub WriteToStream()
      Dim objStreamWriter As StreamWriter
      Dim intA As Integer=100
      Dim intB As Integer=200
      Dim blnSampleBoolean AsBoolean
      Dim dSampleDecimal As Decimal
Try
ObjstreamWriter=File.CreateText("c:\newfile.txt"
      )
With objStreamWriter
.writeline("log file for "&
      FormatDateTime(Now()))
.writeline("-----------------------------------------------")
.writeline("This is a formatted string:{0} {1} ",
      intA,  intB)    .writeline()
.writeline(blnSampleBoolean)
.writeline(dSampleDecimal)
.Close()
End with
Catch E As Exception
      Console.WritLine("Error
      occurred:&"E.Message)
End try
End sub

        Sub Main()
              WriteToStream()
        End sub
      End module
```

output

```
Log File for 7/17/2001 8:58:50 P M
-----------------------------------------------
This is a formatted String : 100 200
True
3.14
```

Using Binary File I/O with the FileStream Object

- Graphic images contain binary data
- BinaryReader and BinaryWriter
- Reading Binary Files
- Writing Binary Files
- For eg., ID3vl information (datablock of 128 bytes
  - song,title,artist,album,year of publication,generation and additional comments

Performing File opeartion

- File class is static (no constructor)
- FileInfo class (retrieve characteristics about file, has constructor)
- File attributes and Archive flag used to backup

```
Imports System
Imports System.IO
Module Module1
Sub DisplayFileInformation(ByVal  Filespec
    As String)
Dim strFileInfo As String
Dim fabfilesAttribs  As New FileAttributes()
Dim objFileOnfo As new FileInfo(FileSpec)
Try
    If fabFileAttribs .Exists Then
    fabFileAttribs = objFileInfo.Attributes
    If fab FileAttributes  And
    FileAttributes.Archieve Then
    Console.WriteLine("Archieve flag set")
    Else
    Console.WriteLine("Archieve flag set")
    End If
strFileInfo = "File Information: {0 } "& _
ControlChars.CrLf & _
"size: {1} " & ControlChars.CrLf&_
"Created: {2} "  & ControlChars,CrLf
    Console.WriteLine(strFileInfo,
    objFileInfo,Name, objFileInfo,Length,
    FormatDateTime(
    objFileInfo.CreationTime))
End If
Catch E As Exception
    Console.WriteLine("ERROR:" &
    E.Message)
End Try
End sub

Sub Main()
DisplayFileInformation("'test,txt")
End Sub
End Module
```

Copying, Moving and Renaming Files

```
Sub CopyAndRename(ByVal FileToCopy As String)
Dim objFileInfo As FileInfo
Try
objFileInfo = New FileInfo(FileToCopy)
objFileInfo.CopyTo(odjFileOnfo.DirectoryName &_
        “Copy of” & objFileInfo,Name,_
         True)
objFileInfo.MoveTo(objFileInfo.DirectoryName &_
         “OLD_” & objFileInfo.Name)
Catch E As Exception
Console.writeLine(“ERROR;”& E.Message)
End Try
End Sub
```