****Startup.vb****

```
'RIC Mobile Course Catalog -- Administrative Tools
'By Lianne Elsner -> 2005 - 2006 Computer Science Honors Project
'~~~~~~~~~~~Startup.vb~~~~~~~~~~~~~~~~
'Creates objects of all forms, runs the splash screen, and transitions
to the main menu
Option Explicit On
Option Strict On

Module ModuleStartup
    Sub main()
        'switch between splash screen and main menu (v0.1)
        Dim frmSplash As New SplashScreen
        Dim frmMain As New MainMenu
        frmSplash.ShowDialog()
        frmMain.ShowDialog()
    End Sub
End Module
```

****SplashScreen.vb****

```
'RIC Mobile Course Catalog - Administrative Tools
'By Lianne Elsner -> 2005 - 2006 Computer Science Honors Project
'~~~~~~~~~~~Splash Screen~~~~~~~~~~~~~~~~
'Shows a splash screen upon start of program
Public Class SplashScreen
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

    Private Sub tmrSplash_Tick(ByVal sender As Object, ByVal e As
System.EventArgs) Handles tmrSplash.Tick
        'close the form when the timer has reached the end (approx. 5
seconds) (v0.1)
        Me.Close()
    End Sub
End Class
```

****MainMenu.vb****

```
'RIC Mobile Course Catalog - Administrative Tools
'By Lianne Elsner -> 2005 - 2006 Computer Science Honors Project
'~~~~~~~~~~~Main Menu~~~~~~~~~~~~~~~~
'Main interface for editing the database
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''Version Tracking Info''''''''''''''''''''''''''
'''''''Alpha Versions'''''''''''''''
'Version 0.1 -- 10/12/2005 -- added functionality to all buttons,
allowing user to navigate
' through entire program, neatened form design, created Startup.vb
module, set up splash screen
'Version 0.2 -- 10/23/2005 -- set up 'Edit Title' frame, write code to
update changes made to database
'Version 0.3 -- 10/25/2005 -- set up radio buttons to enable and
disable controls on forms, added dialog boxes for cancelling
'edit
```

```
'Version 0.4 -- 10/28/2005 -- set up Majors form to read from database,
write code to add, edit, and delete majors
'Version 0.5 -- 11/3/2005 -- set up Subcategories form to read from
database, write code to add, edit,
'and delete subcategories, put primary keys of majors into an array
'Version 0.6 -- 11/22/2005 -- set up Courses form to read from
database, write code to add, edit, and delete courses
''''''''Beta Versions'''''''''''''''''
'Version 1.0 -- 1/3/2006 -- Remove orphaned subcategories when major is
deleted.
'Version 1.1 -- 1/18/2006 -- Delete partnership between deleted
subcategories and courses
'Version 1.2 -- 1/25/2006 -- Made array sizes variable instead of fixed
'Version 1.3 -- 1/30/2005 -- Added background image to Menu
'Version 1.4 -- 2/4/2006 -- Added error checking to make sure the
mimimum amount of data is properly entered before
'the data is submitted to the database
''''''''Final Release'''''''''''''''''
'Version 1.4 - 2/4/2006 -- Fully tested, fully functional.

Option Explicit On
Option Strict On

Imports System.Data
Imports System.Data.Common
Imports System.Data.OleDb
Imports System.Convert


Public Class MainMenu
    Inherits System.Windows.Forms.Form

#Region " Windows Form Designer generated code "

    Private dtMajors As DataTable
    Private dtCourses As DataTable
    Private dtSubCats As DataTable
    'arrays to hold the primary keys of items in the dropdown lists
(v0.5)
    'array size changes dynamically based on the number of rows in the
table at a given time (v1.2)
    Private SubCatArray(1) As Integer 'holds an array of the
subcategories' primary keys
    Private CourseArray(1) As Integer 'holds an array of the courses'
primary keys
    Private MajorsArray(1) As Integer 'hold an array of the majors'
primary keys

    Private Sub MainMenu_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'fill the data adaptors (v0.2)
        daCourses.Fill(Ds1)
        daMajors.Fill(Ds1)
        daSubCat.Fill(Ds1)
        daTitle.Fill(Ds1)
        dtMajors = Ds1.Tables("Majors")
        dtCourses = Ds1.Tables("Courses")
```

```
        dtSubCats = Ds1.Tables("Subcategories")
        rbNewCourse.Select()
        rbNewMajor.Select()
        rbNewSubcat.Select()
    End Sub

    Private Sub MainMenu_Closing(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
        'close connection to the database when the program is closed
(v0.2)
        conn.Close()
    End Sub

    '-------------------------Buttons--------------------------------
    Private Sub btnEditMajors_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnEditMajors.Click
        fraMajors.Visible = True
        fraSubCat.Visible = False
        fraCourses.Visible = False
        fraTitle.Visible = False
        pbCatalogCover.Visible = False
        'Disable buttons - i don't want the user to click them at this
point.
        btnEditMajors.Enabled = False
        btnEditSubcat.Enabled = False
        btnEditCourses.Enabled = False
        btnTitle.Enabled = False
        btnExit.Enabled = False
        rbNewMajor.Checked = True
        Call PopulateMajors() 'populates the Majors dropdown box (v0.4)
    End Sub

    Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnExit.Click
        Me.Close()
    End Sub

    Private Sub btnEditSubcat_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnEditSubcat.Click
        fraMajors.Visible = False
        fraSubCat.Visible = True
        fraCourses.Visible = False
        fraTitle.Visible = False
        pbCatalogCover.Visible = False
        'Disable buttons - i don't want the user to click them at this
point.
        btnEditMajors.Enabled = False
        btnEditSubcat.Enabled = False
        btnEditCourses.Enabled = False
        btnTitle.Enabled = False
        btnExit.Enabled = False
        Call ResetSubcatListbox() 'clears the listboxes (v0.5)
        Call PopulateSubcatMajor() 'populates the Majors listbox (v0.5)
    End Sub

    Private Sub btnEditCourses_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnEditCourses.Click
```

```
        fraMajors.Visible = False
        fraSubCat.Visible = False
        fraCourses.Visible = True
        fraTitle.Visible = False
        pbCatalogCover.Visible = False
        'Disable buttons - i don't want the user to click them at this
point.
        btnEditMajors.Enabled = False
        btnEditSubcat.Enabled = False
        btnEditCourses.Enabled = False
        btnTitle.Enabled = False
        btnExit.Enabled = False
        Call PopulateCourses() 'populates the Courses dropdown box
(v0.6)
        Call PopulateTree() 'populates the treeview of avaliable
courses and subcategories (v0.6)
    End Sub

    Private Sub btnTitle_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnTitle.Click
        fraMajors.Visible = False
        fraSubCat.Visible = False
        fraCourses.Visible = False
        fraTitle.Visible = True
        pbCatalogCover.Visible = False
        'Disable buttons - i don't want the user to click them at this
point.
        btnEditMajors.Enabled = False
        btnEditSubcat.Enabled = False
        btnEditCourses.Enabled = False
        btnTitle.Enabled = False
        btnExit.Enabled = False
        '''Put old title in text box for editing (v0.2)''''
        Try
            Dim dRow As ds.TitleRow
            dRow = Ds1.Title(0)
            txtNewTitle.Text = CStr(dRow.Catalog_Title)
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    Private Sub rbEditMajor_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbEditMajor.CheckedChanged
        ''Enable/disable controls (v0.3)
        txtNewMajor.Enabled = True
        cmbOldMajor.Enabled = True
    End Sub

    Private Sub rbDeleteMajor_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbDeleteMajor.CheckedChanged
        ''Enable/disable controls (v0.3)
        txtNewMajor.Enabled = False
        cmbOldMajor.Enabled = True
    End Sub
```

```
    Private Sub rbNewMajor_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbNewMajor.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbOldMajor.Enabled = False
        txtNewMajor.Enabled = True
    End Sub

    Private Sub rbEditSubcat_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbEditSubcat.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbOldSubcat.Enabled = True
        txtNewSubcat.Enabled = True
        cmbSubcatMajor.Enabled = True
    End Sub

    Private Sub rbDeleteSubcat_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbDeleteSubcat.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbOldSubcat.Enabled = True
        txtNewSubcat.Enabled = False
        cmbSubcatMajor.Enabled = True
    End Sub

    Private Sub rbNewSubcat_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbNewSubcat.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbOldSubcat.Enabled = False
        txtNewSubcat.Enabled = True
        cmbSubcatMajor.Enabled = True
    End Sub

    Private Sub rbEditCourse_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbEditCourse.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbNewCourse.Enabled = True
        txtNewCourse.Enabled = True
        txtCoursePrereq.Enabled = True
        txtCourseCode.Enabled = True
        txtCourseCredits.Enabled = True
        txtCourseTerm.Enabled = True
        txtCourseDescription.Enabled = True
        trvCourseSubcat.Enabled = True
    End Sub

    Private Sub rbDeleteCourse_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbDeleteCourse.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbNewCourse.Enabled = True
        txtNewCourse.Enabled = False
        txtCoursePrereq.Enabled = False
```

```
        txtCourseCredits.Enabled = False
        txtCourseTerm.Enabled = False
        txtCourseCode.Enabled = False
        txtCourseDescription.Enabled = False
        trvCourseSubcat.Enabled = False
    End Sub

    Private Sub rbNewCourse_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
rbNewCourse.CheckedChanged
        ''Enable/disable controls (v0.3)
        cmbNewCourse.Enabled = False
        txtNewCourse.Enabled = True
        txtCoursePrereq.Enabled = True
        txtCourseCode.Enabled = True
        txtCourseCredits.Enabled = True
        txtCourseTerm.Enabled = True
        txtCourseDescription.Enabled = True
        trvCourseSubcat.Enabled = True
    End Sub

    '---------------Handle Majors Widget-----------------------------
-------
    Private Sub PopulateMajors()
        Call ResetArray(MajorsArray) 'reset the array of majors(v0.5)
        'Reset input boxes for next action (v0.4)
        Dim intItemCount As Integer
        intItemCount = 0
        While intItemCount < cmbOldMajor.Items.Count
            cmbOldMajor.Items.RemoveAt(intItemCount)
        End While
        cmbOldMajor.Text = "Select Major"
        txtNewMajor.Clear()
        'Populate dropdown list with Majors from database (v0.4)
        Try
            Dim dRow As ds.MajorsRow
            Dim intCounter As Integer
            intCounter = 0
            Dim index As Integer = 0
            'code to make array size variable (v1.2)
            Dim arraysize As Integer
            arraysize = Ds1.Majors.Count
            ReDim MajorsArray(arraysize)
            For Each dRow In Ds1.Majors

cmbOldMajor.Items.Add(CStr(Ds1.Majors(intCounter).Major))
                MajorsArray.SetValue(Ds1.Majors(intCounter).Major_Code,
index)
                index = index + 1
                intCounter = intCounter + 1
            Next
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub
```

```
    Private Sub btnSubmitMajor_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSubmitMajor.Click
        Dim valResult As String
        'Make changes to the selected Major (v0.4)
        Try
            Dim dRow As ds.MajorsRow
            Dim currMajorIndex As Integer
            Dim ChangeIndex As Integer
            Dim currMajorKey As Integer
            currMajorIndex = cmbOldMajor.SelectedIndex
            If rbEditMajor.Checked = True Then
                valResult = ValidateInput("majorEdit")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'save changes made to selected major
                ChangeIndex = 0
                currMajorKey =
CInt(MajorsArray.GetValue(currMajorIndex))
                For Each dRow In Ds1.Majors
                    If Ds1.Majors(ChangeIndex).Major_Code =
currMajorKey Then
                        dRow = Ds1.Majors(ChangeIndex)
                        dRow.BeginEdit() 'lock records
                        dRow.Major = txtNewMajor.Text()
                        dRow.EndEdit() 'unlock records
                        Call SaveChanges(daMajors)
                        MessageBox.Show("Major has been changed.",
"Success")
                        Exit Try 'major has been found - exit this part
of the subroutine
                    End If
                    ChangeIndex = ChangeIndex + 1
                Next
            ElseIf rbNewMajor.Checked = True Then
                valResult = ValidateInput("majorAdd")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'add new major to database
                Dim dr As DataRow
                dr = dtMajors.NewRow
                dr("Major") = txtNewMajor.Text
                dtMajors.Rows.Add(dr)
                Call SaveChanges(daMajors)
                MessageBox.Show("Major has been added.", "Success")
                ElseIf rbDeleteMajor.Checked = True Then
                valResult = ValidateInput("majorDel")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
```

```
                'delete selected major
                Dim dr As DataRow
                Dim result As DialogResult
                result = MessageBox.Show("Are you sure you wish to
delete this major?", "Delete", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
                If result = DialogResult.Yes Then
                    currMajorKey =
CInt(MajorsArray.GetValue(currMajorIndex))
                    ChangeIndex = 0
                    For Each dRow In Ds1.Majors
                        If Ds1.Majors(ChangeIndex).Major_Code =
currMajorKey Then
                            dr = dtMajors.Rows(ChangeIndex)
                            dr.Delete()
                            Call SaveChanges(daMajors)
                            Call RemoveDeadSubcat(currMajorKey)
                            MessageBox.Show("Major has been deleted.",
"Success")
                            Exit Try 'escape to prevent error caused by
incrementing changeindex after deletion
                        End If
                        ChangeIndex = ChangeIndex + 1
                    Next
                End If
                End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
        'Allow user to add another major or return to Main Menu (v0.2)
        Dim response As DialogResult
        response = MessageBox.Show("Would you like to edit another
major?", "Administrative Tools", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
        If response = DialogResult.No Then
            fraMajors.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
        Else
            'Refresh the list of majors (v0.4)
            Call PopulateMajors()
        End If
    End Sub

    Private Sub btnCancelMajor_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCancelMajor.Click
        Dim response As DialogResult
        response = MessageBox.Show("Cancel changes made to Major?",
"Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
        If response = DialogResult.Yes Then
            fraMajors.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
```

```
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
            'reset majors dropdown
            Call PopulateMajors()
        End If
    End Sub

    '------------------Handle Subcategories Widget----------------------
---
    Private Sub PopulateSubcatMajor()
        Call ResetArray(MajorsArray) 'reset the array of majors(v0.5)
        'Reset input boxes for next action (v0.5)
        Dim intItemCount As Integer
        intItemCount = 0
        While intItemCount < cmbSubcatMajor.Items.Count
            cmbSubcatMajor.Items.RemoveAt(intItemCount)
        End While
        cmbSubcatMajor.Text = "Select Major"
        txtNewSubcat.Clear()
        'Populate dropdown list with Majors from database (v0.5)
        Try
            Dim dRow As ds.MajorsRow
            Dim intCounter As Integer
            Dim index As Integer = 0
            intCounter = 0
            'code to make array size variable (v1.2)
            Dim arraysize As Integer
            arraysize = Ds1.Majors.Count
            ReDim MajorsArray(arraysize)
            For Each dRow In Ds1.Majors

cmbSubcatMajor.Items.Add(CStr(Ds1.Majors(intCounter).Major))
                MajorsArray.SetValue(Ds1.Majors(intCounter).Major_Code,
index)
                intCounter = intCounter + 1
                index = index + 1
            Next
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    Private Sub btnSubmitSubcat_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSubmitSubcat.Click
        Dim valResult As String
        'Make changes to the selected subcategory (v0.5)
        Try
            Dim dRow As ds.SubcategoriesRow
            Dim currSubCatIndex As Integer 'holds the array index of
the selected subcategory
            Dim currSubcatKey As Integer 'holds the primary key of the
selected subcategory
            Dim currMajorKey As Integer  'holds the primary key of the
selected subcategory
```

```
            Dim currMajorIndex As Integer 'holds the array index of the
selected subcategory
            Dim ChangeIndex As Integer  'holds the database index of
the selected subcategory
            currSubCatIndex = cmbOldSubcat.SelectedIndex
            currMajorIndex = cmbSubcatMajor.SelectedIndex
            If rbEditSubcat.Checked = True Then
                valResult = ValidateInput("subcatEdit")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'save changes made to selected subcategory
                ChangeIndex = 0
                currSubcatKey =
CInt(SubCatArray.GetValue(currSubCatIndex))
                For Each dRow In Ds1.Subcategories
                    If Ds1.Subcategories(ChangeIndex).SubCatCode =
currSubcatKey Then
                        dRow = Ds1.Subcategories(ChangeIndex)
                        dRow.BeginEdit() 'lock records
                        dRow.SubCatName = txtNewSubcat.Text()
                        dRow.EndEdit() 'unlock records
                        Call SaveChanges(daSubCat)
                        MessageBox.Show("Subcategory has been
changed.", "Success")
                        Exit Try 'item found - exit this part of the
subroutine
                    End If
                    ChangeIndex = ChangeIndex + 1
                Next
            ElseIf rbNewSubcat.Checked = True Then
                valResult = ValidateInput("subcatAdd")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                ChangeIndex = 0
                currMajorKey =
CInt(MajorsArray.GetValue(currMajorIndex))
                For Each dRow In Ds1.Subcategories
                    If Ds1.Majors(ChangeIndex).Major_Code =
currMajorKey Then
                        Dim dr As DataRow
                        dr = dtSubCats.NewRow
                        dr("SubCatName") = txtNewSubcat.Text
                        dr("CorrMajorCode") = currMajorKey
                        dtSubCats.Rows.Add(dr)
                        Call SaveChanges(daSubCat)
                        MessageBox.Show("Subcategory has been added.",
"Success")
                        Exit Try 'item found - exit this part of the
subroutine
                    End If
                    ChangeIndex = ChangeIndex + 1
```

```
                Next
            ElseIf rbDeleteSubcat.Checked = True Then
                valResult = ValidateInput("subcatDel")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'delete selected subcategory
                Dim dr As DataRow
                Dim result As DialogResult
                result = MessageBox.Show("Are you sure you wish to
delete this subcategory?", "Delete", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
                If result = DialogResult.Yes Then
                    currSubcatKey =
CInt(SubCatArray.GetValue(currSubCatIndex))
                    ChangeIndex = 0
                    For Each dRow In Ds1.Subcategories
                        If Ds1.Subcategories(ChangeIndex).SubCatCode =
currSubcatKey Then
                            dr = dtSubCats.Rows(ChangeIndex)
                            Call
RemoveDeadSubcatFromCourse(Ds1.Subcategories(ChangeIndex).SubCatCode)
'(v1.1)
                            dr.Delete()
                            Call SaveChanges2(daSubCat, daCourses)
                            MessageBox.Show("Subcategory has been
deleted.", "Success")
                            Exit Try 'prevents an error caused by
incrementing ChangeIndex after element is deleted (v0.5)
                        End If
                        ChangeIndex = ChangeIndex + 1
                    Next
                End If
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
        Dim response As DialogResult
        response = MessageBox.Show("Would you like to edit another
subcategory?", "Administrative Tools", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
        If response = DialogResult.No Then
            fraSubCat.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
        End If
        'reset listboxes
        Call ResetSubcatListbox()
        Call PopulateSubcatMajor()
    End Sub
```

```
    Private Sub btnCancelSubcat_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCancelSubcat.Click
        Dim response As DialogResult
        response = MessageBox.Show("Cancel changes made to
Subcategory?", "Are you sure?", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
        If response = DialogResult.Yes Then
            fraSubCat.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
            'reset listboxes
            Call PopulateSubcatMajor()
            Call ResetSubcatListbox()
        End If
    End Sub

    Private Sub cmbSubcatMajor_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
cmbSubcatMajor.SelectedIndexChanged
        'populate subcategories dropdown based on which major has been
selected (v0.5)
        Call ResetSubcatListbox()
        Dim dRow As ds.MajorsRow
        Dim currMajor As Integer
        Call ResetArray(SubCatArray)
        currMajor = cmbSubcatMajor.SelectedIndex
        dRow = Ds1.Majors(currMajor)
        Try
            Dim dSRow As ds.SubcategoriesRow
            Dim intCounter As Integer = 0
            Dim index As Integer = 0
            'code to make array size variable (v1.2)
            Dim arraysize As Integer
            arraysize = Ds1.Subcategories.Count
            ReDim SubCatArray(arraysize)
            For Each dSRow In Ds1.Subcategories
                If Ds1.Majors(currMajor).Major_Code =
Ds1.Subcategories(intCounter).CorrMajorCode Then

cmbOldSubcat.Items.Add(Ds1.Subcategories(intCounter).SubCatName)

SubCatArray.SetValue(Ds1.Subcategories(intCounter).SubCatCode, index)
                    index = index + 1
                End If
                intCounter = intCounter + 1
            Next
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    Private Sub ResetSubcatListbox()
        'reset the listbox for new data (v0.5)
```

```
        Dim intItemCount As Integer
        intItemCount = 0
        While intItemCount < cmbOldSubcat.Items.Count
            cmbOldSubcat.Items.RemoveAt(intItemCount)
        End While
        cmbOldSubcat.Text = "Select Subcategory"
    End Sub

    '-------------Handle Title Widget-------------------------
    Private Sub btnTitleSubmit_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnTitleSubmit.Click
        If txtNewTitle.Text = "" Then 'error checking (v1.4)
            MessageBox.Show("Please enter a new title.", "Error")
        Else
            fraTitle.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
            '''Submit changed title to database (v0.2)'''
            Try
                Dim dRow As ds.TitleRow
                dRow = Ds1.Title(0)
                dRow.BeginEdit() 'lock records
                dRow.Catalog_Title = txtNewTitle.Text()
                dRow.EndEdit() 'unlock records
                Call SaveChanges(daTitle)
                MessageBox.Show("Title has been changed.", "Success")
            Catch ex As Exception
                MessageBox.Show(ex.Message, "Error")
            End Try
        End If
    End Sub

    Private Sub btnTitleCancel_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnTitleCancel.Click
        Dim response As DialogResult
        response = MessageBox.Show("Cancel changes made to Title?",
"Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
        If response = DialogResult.Yes Then
            fraTitle.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
        End If
    End Sub

    '--------------Handle Courses Widget--------------------
    Private Sub btnSubmitCourse_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSubmitCourse.Click
        Dim valResult As String
        'update the selected course in the database (v0.6)
```

```
        Try
            Dim dRow As ds.CoursesRow
            Dim currCourseIndex As Integer 'holds the array index of
the selected course
            Dim currCourseKey As Integer 'holds the primary key of the
selected course
            Dim ChangeIndex As Integer  'holds the database index of
the selected course
            Dim subcatList As String 'holds the list of selected
subcategories
            Dim rootNode As TreeNode 'to be passed to the function that
grabs the subcats from the tree
            rootNode = ReturnRoot()
            currCourseIndex = cmbNewCourse.SelectedIndex
            If rbEditCourse.Checked = True Then
                valResult = ValidateInput("courseEdit")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'save changes made to selected course
                currCourseKey =
CInt(CourseArray.GetValue(currCourseIndex))
                ChangeIndex = 0
                For Each dRow In Ds1.Courses
                    If Ds1.Courses(ChangeIndex).Course_Key =
currCourseKey Then
                        dRow = Ds1.Courses(ChangeIndex)
                        dRow.BeginEdit() 'lock records
                        dRow.Course_Name = txtNewCourse.Text()
                        dRow.Course_Code = txtCourseCode.Text
                        dRow.PreRequisites = txtCoursePrereq.Text
                        dRow.Number_of_Credits = txtCourseCredits.Text
                        dRow.Semesters_Offered = txtCourseTerm.Text
                        dRow.Description = txtCourseDescription.Text
                        subcatList = UpdateSubcats(rootNode, "")
                        dRow.CorrSubcatCode = subcatList
                        dRow.EndEdit() 'unlock records
                        Call SaveChanges(daCourses)
                        MessageBox.Show("Course has been changed.",
"Success")
                        Exit Try 'item found - exit this part of the
subroutine
                    End If
                    ChangeIndex = ChangeIndex + 1
                Next
            ElseIf rbNewCourse.Checked = True Then
                valResult = ValidateInput("courseAdd")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                'add new course to database
                Dim dr As DataRow
                dr = dtCourses.NewRow
```

```
                dr("Course Name") = txtNewCourse.Text
                dr("Course_Code") = txtCourseCode.Text
                dr("Number of Credits") = txtCourseCredits.Text
                dr("Semesters Offered") = txtCourseTerm.Text
                dr("Prerequisites") = txtCoursePrereq.Text
                dr("Description") = txtCourseDescription.Text
                subcatList = UpdateSubcats(rootNode, "")
                dr("CorrSubcatCode") = subcatList
                dtCourses.Rows.Add(dr)
                Call SaveChanges(daCourses)
                MessageBox.Show("Course has been added.", "Success")
                Exit Try 'item found - exit this part of the subroutine
            ElseIf rbDeleteCourse.Checked = True Then
                valResult = ValidateInput("courseDel")
                'if validations fails, break out of subroutine
                If Not valResult.Equals("ok") Then
                    MessageBox.Show(valResult, "Error")
                    Exit Sub
                End If
                currCourseKey =
CInt(CourseArray.GetValue(currCourseIndex))
                'delete selected course
                Dim dr As DataRow
                Dim result As DialogResult
                result = MessageBox.Show("Are you sure you wish to
delete this course?", "Delete", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
                If result = DialogResult.Yes Then
                    ChangeIndex = 0
                    For Each dRow In Ds1.Courses
                        If Ds1.Courses(ChangeIndex).Course_Key =
currCourseKey Then
                            dr = dtCourses.Rows(ChangeIndex)
                            dr.Delete()
                            Call SaveChanges(daCourses)
                            MessageBox.Show("Course has been deleted.",
"Success")
                            Exit Try 'prevents an error caused by
incrementing ChangeIndex after element is deleted (v0.5)
                        End If
                        ChangeIndex = ChangeIndex + 1
                    Next
                End If
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
        Dim response As DialogResult
        response = MessageBox.Show("Would you like to edit another
course?", "Administrative Tools", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
        If response = DialogResult.No Then
            fraCourses.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
```

```
            btnTitle.Enabled = True
            btnExit.Enabled = True
        End If
        'reset listboxes
        Dim tn As TreeNode
        tn = ReturnRoot()
        Call PopulateCourses()
        Call UncheckAllNodes(tn)
    End Sub

    Private Sub btnCancelCourse_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCancelCourse.Click
        'cancel edit (v0.6)
        Dim response As DialogResult
        response = MessageBox.Show("Cancel changes made to Course?",
"Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
        If response = DialogResult.Yes Then
            fraCourses.Visible = False
            pbCatalogCover.Visible = True
            btnEditMajors.Enabled = True
            btnEditSubcat.Enabled = True
            btnEditCourses.Enabled = True
            btnTitle.Enabled = True
            btnExit.Enabled = True
            'reset listboxes
            Dim tn As TreeNode
            tn = ReturnRoot()
            Call PopulateCourses()
            Call UncheckAllNodes(tn)
        End If
    End Sub

    Private Sub PopulateCourses()
        'populate the courses dropdown
        Call ResetArray(CourseArray) 'reset the array of majors(v0.5)
        'clear all textboxes
        txtNewCourse.Clear()
        txtCourseCode.Clear()
        txtCoursePrereq.Clear()
        txtCourseCredits.Clear()
        txtCourseTerm.Clear()
        txtCourseDescription.Clear()
        'clear the courses dropdown
        Dim intItemCount As Integer
        intItemCount = 0
        While intItemCount < cmbNewCourse.Items.Count
            cmbNewCourse.Items.RemoveAt(intItemCount)
        End While
        cmbNewCourse.Text = "Select Course"
        'populate courses dropdown from the database
        Try
            Dim dRow As ds.CoursesRow
            Dim intCounter As Integer
            Dim index As Integer = 0
            intCounter = 0
            'code to make array size variable (v1.2)
            Dim arraysize As Integer
```

```
            arraysize = Ds1.Courses.Count
            ReDim CourseArray(arraysize)
            For Each dRow In Ds1.Courses

cmbNewCourse.Items.Add(CStr(Ds1.Courses(intCounter).Course_Code))

CourseArray.SetValue(Ds1.Courses(intCounter).Course_Key, index)
                intCounter = intCounter + 1
                index = index + 1
            Next
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    Private Sub cmbNewCourse_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
cmbNewCourse.SelectedIndexChanged
        Dim dCRow As ds.CoursesRow
        Dim dSRow As ds.SubcategoriesRow
        Dim currCourse As Integer
        Dim subcatsToSplit As String
        Dim subcatList() As String
        Dim arrayCounter, dbCounter As Integer
        Dim size As Integer
        Dim delim() As Char = {ToChar(",")}
        Dim tn As TreeNode
        tn = ReturnRoot()
        Call UncheckAllNodes(tn)
        currCourse = cmbNewCourse.SelectedIndex
        Try
            dCRow = Ds1.Courses(currCourse)
            txtNewCourse.Text = dCRow.Course_Name
            txtCourseCode.Text = dCRow.Course_Code
            txtCoursePrereq.Text = dCRow.PreRequisites
            txtCourseCredits.Text = dCRow.Number_of_Credits
            txtCourseTerm.Text = dCRow.Semesters_Offered
            txtCourseDescription.Text = dCRow.Description
            subcatsToSplit = dCRow.CorrSubcatCode
            subcatList = subcatsToSplit.Split(delim)
            size = subcatList.Length
            dbCounter = 0
            For Each dSRow In Ds1.Subcategories
                dSRow = Ds1.Subcategories(dbCounter)
                For arrayCounter = 0 To size - 1
                    If CStr(dSRow.SubCatCode) =
subcatList(arrayCounter) Then
                        tn = ReturnRoot()
                        Call CheckTreeNode(tn, CStr(dSRow.SubCatCode))
                    End If
                Next
                dbCounter = dbCounter + 1
            Next
        Catch ex As Exception
            'throws the exception if subcategories list is null -
continue with program, do not break
        End Try
```

```
    End Sub

    Private Function ReturnRoot() As TreeNode
        'returns the root node for use in other functions
        Dim tnNode As TreeNode
        tnNode = trvCourseSubcat.Nodes.Item(0)
        Return tnNode
    End Function

    Private Sub CheckTreeNode(ByVal tnNode As TreeNode, ByVal key As
String)
        'recursive subroutine that cycles through the tree and checks
off the node with the given key value
        Dim tag As String
        Dim tn As TreeNode
        For Each tn In tnNode.Nodes
            tag = CStr(tn.Tag)
            If tag = "S," & key Then
                tn.Checked = True
            End If
            CheckTreeNode(tn, key)
        Next
    End Sub

    Private Sub UncheckAllNodes(ByVal tnNode As TreeNode)
        Dim tn As TreeNode
        For Each tn In tnNode.Nodes
            If tn.Checked = True Then
                tn.Checked = False
            End If
            UncheckAllNodes(tn)
        Next
    End Sub

    Private Sub PopulateTree()
        'populates the treeview with avaliable majors and subcategories
(v0.6)
        Dim tnRoot As TreeNode
        Dim title As String
        Try
            Dim titleRow As ds.TitleRow
            titleRow = Ds1.Title(0)
            title = CStr(titleRow.Catalog_Title)
            trvCourseSubcat.Nodes.Clear()
            trvCourseSubcat.BeginUpdate()
            tnRoot = trvCourseSubcat.Nodes.Add(title)
            tnRoot.Tag = "R,"
            trvCourseSubcat.EndUpdate()
            AddMajorNode(tnRoot)
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    Private Sub AddMajorNode(ByVal tnNode As TreeNode)
        'adds majors to the treeview (v0.6)
        Dim tnMajor As TreeNode
```

```
        Dim majorRow As ds.MajorsRow
        Dim row As Integer
        For row = 0 To Ds1.Majors.Rows.Count - 1
            majorRow = Ds1.Majors(row)
            tnMajor = tnNode.Nodes.Add(majorRow.Major)
            tnMajor.Tag = "M," & majorRow.Major_Code
            AddSubcatNode(tnMajor, majorRow)
        Next
    End Sub

    Private Sub AddSubcatNode(ByVal tnNode As TreeNode, ByVal majorRow
As ds.MajorsRow)
        'adds subcategories to the treeview (v0.6)
        Dim subcatRow As ds.SubcategoriesRow
        Dim tnSubcat As TreeNode
        Dim count As Integer
        Dim tagVal As String
        tagVal = GetTagVal(CStr(tnNode.Tag))
        For count = 0 To Ds1.Subcategories.Rows.Count - 1
            subcatRow = Ds1.Subcategories(count)
            If tagVal = CStr(subcatRow.CorrMajorCode) Then
                tnSubcat = tnNode.Nodes.Add(subcatRow.SubCatName)
                tnSubcat.Tag = "S," & subcatRow.SubCatCode
            End If
        Next
    End Sub

    Private Function GetTagVal(ByVal tag As String) As String
        'removes the leading letter and comma from the tag, returns the
rest of the string (v0.6)
        Dim holder As String
        Dim commaIndex As Integer
        commaIndex = tag.IndexOf(",")
        holder = tag.Substring(commaIndex + 1)
        Return holder
    End Function

    Private Sub trvCourseSubcat_AfterCheck(ByVal sender As Object,
ByVal e As System.Windows.Forms.TreeViewEventArgs) Handles
trvCourseSubcat.AfterCheck
        'if user checks off the root node or a major, uncheck the box
(only subcategories can be selected) (v0.6)
        Dim tag As String
        Dim tagLetter As String
        tag = CStr(e.Node.Tag)
        tagLetter = tag.Substring(0, 1)
        If Not tagLetter.Equals("S") Then
            If e.Node.Checked Then 'uncheck if it is checked (required
for error handling) (v0.6)
                e.Node.Checked = False
            End If
        End If
    End Sub

    Private Function UpdateSubcats(ByVal tNode As TreeNode, ByVal
subcatString As String) As String
```

```
        'reads through the tree and compiles a comma-delimited list of
the keys of all checked nodes
        Dim childNode As TreeNode
        For Each childNode In tNode.Nodes
            If childNode.Checked = True Then
                subcatString = subcatString &
GetTagVal(CStr(childNode.Tag)) & ","
            End If
            subcatString = UpdateSubcats(childNode, subcatString)
        Next
        Return subcatString
    End Function

    'Deletes subcategories when corresponding major is deleted (v1.0)
    Private Sub RemoveDeadSubcat(ByVal majorCode As Integer)
        Try
            Dim dRow As ds.SubcategoriesRow
            Dim intCounter As Integer
            Dim dr As DataRow
            intCounter = 0
            For Each dRow In Ds1.Subcategories
                If Ds1.Subcategories(intCounter).CorrMajorCode =
majorCode Then
                    dr = dtSubCats.Rows(intCounter)
                    Call
RemoveDeadSubcatFromCourse(Ds1.Subcategories(intCounter).SubCatCode)
'(v1.1)
                    dr.Delete()
                End If
                intCounter = intCounter + 1
            Next
            Call SaveChanges2(daSubCat, daCourses)
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub

    'Deletes dead subcategories from course listings
    Private Sub RemoveDeadSubcatFromCourse(ByVal subcatCode As Integer)
        Dim dRow As ds.CoursesRow
        Dim currCourse As Integer
        Dim subcatsToSplit As String
        Dim subcatList() As String
        Dim size As Integer
        Dim delim() As Char = {ToChar(",")}
        Dim counter As Integer
        Dim newSubcatList As String = ""
        Try
            For Each dRow In Ds1.Courses
                subcatsToSplit = dRow.CorrSubcatCode
                subcatList = subcatsToSplit.Split(delim)
                size = subcatList.Length
                For counter = 0 To size - 1
                    If CStr(subcatCode) = subcatList(counter) Then
                        'remove the item
                        newSubcatList = newSubcatList
                    ElseIf subcatList(counter) = "" Then
```

```
                        'catches last item, prevents string of commas
at the end of the list
                        newSubcatList = newSubcatList &
subcatList(counter)
                    Else
                        'keep the item
                        newSubcatList = newSubcatList &
subcatList(counter) & ","
                    End If
                Next
                dRow.BeginEdit() 'lock records
                dRow.CorrSubcatCode = newSubcatList
                dRow.EndEdit() 'unlock records
                newSubcatList = ""
            Next
        Catch ex As Exception
            'throws the exception if subcategories list is null -
continue with program, do not break
        End Try
    End Sub

    '---------------Shared Methods---------------------
    Private Sub SaveChanges(ByVal da As DataAdapter)
        'Write the changes to the database (v0.2)
        Dim dsUpdates As DataSet
        Try
            dsUpdates = Ds1.GetChanges
            If Not (dsUpdates Is Nothing) Then
                da.Update(dsUpdates)
                Ds1.AcceptChanges()
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub
    Private Sub SaveChanges2(ByVal da1 As DataAdapter, ByVal ds2 As
DataAdapter)
        'Used when 2 data adapters need to be updated at the same time
(v1.1)
        Dim dsUpdates As DataSet
        Try
            dsUpdates = Ds1.GetChanges
            If Not (dsUpdates Is Nothing) Then
                da1.Update(dsUpdates)
                ds2.Update(dsUpdates)
                Ds1.AcceptChanges()
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message, "Error")
        End Try
    End Sub
    Private Sub ResetArray(ByVal a() As Integer)
        'reset the array of subcategory primary keys (v0.5)
        Dim ArrayLength As Integer = a.Length
        Array.Clear(a, 0, ArrayLength)
    End Sub
```

```
    'verify that the minimum amount of data has been entered for
processing.  (v1.4)
    Private Function ValidateInput(ByVal type As String) As String
        Dim returnResult As String = "ok" 'if none of the below return
errors, return "ok"
        If type.Equals("majorAdd") Then
            If txtNewMajor.Text = "" Then
                returnResult = "Please enter a name for the new major."
            End If
        ElseIf type.Equals("majorEdit") Then
            If cmbOldMajor.SelectedIndex < 0 Then
                returnResult = "Please select a major from the list to
edit."
            ElseIf txtNewMajor.Text = "" Then
                returnResult = "Please enter a name for the new major."
            End If
        ElseIf type.Equals("majorDel") Then
            If cmbOldMajor.SelectedIndex < 0 Then
                returnResult = "Please select a major from the list to
delete."
            End If
        ElseIf type.Equals("subcatAdd") Then
            If cmbSubcatMajor.SelectedIndex < 0 Then
                returnResult = "Please select a major from the list."
            ElseIf txtNewSubcat.Text = "" Then
                returnResult = "Please enter a name for the new
subcategory."
            End If
        ElseIf type.Equals("subcatEdit") Then
            If cmbSubcatMajor.SelectedIndex < 0 Then
                returnResult = "Please select a major from the list."
            ElseIf cmbOldSubcat.SelectedIndex < 0 Then
                returnResult = "Please select a subcategory from the
list to edit."
            ElseIf txtNewSubcat.Text = "" Then
                returnResult = "Please enter a name for the new
subcategory."
            End If
        ElseIf type.Equals("subcatDel") Then
            If cmbSubcatMajor.SelectedIndex < 0 Then
                returnResult = "Please select a major from the list."
            ElseIf cmbOldSubcat.SelectedIndex < 0 Then
                returnResult = "Please select a subcategory from the
list to delete."
            End If
        ElseIf type.Equals("courseAdd") Then
            If txtCourseCode.Text = "" Then
                returnResult = "Please enter a course code for the new
course."
            End If
        ElseIf type.Equals("courseEdit") Then
            If cmbNewCourse.SelectedIndex < 0 Then
                returnResult = "Please select a course from the list to
edit."
            ElseIf txtCourseCode.Text = "" Then
                returnResult = "Please enter a course code for the new
course."
```

```
            End If
        ElseIf type.Equals("courseDel") Then
            If cmbNewCourse.SelectedIndex < 0 Then
                returnResult = "Please select a course from the list to
delete."
            End If
        End If
        Return returnResult
    End Function
End Class
```