Lab 3
The High-Low Game

LAB GOALS

To develop a simple windows-based game named “High-Low” using VB .Net.

You will use:  Buttons, Textboxes, Labels, Dim, integer, arithmetic operations, conditionals [if-then-else], random numbers [Randomize(), and Rnd() functions], type conversion functions [CInt (), CStr], Compiler directives such as Option Strict, Option Explicit, input validation and error checking [IsNumeric() function].

Step 1:  Create a new project named HighLow. Once the main form is displayed, create the following interface. Make sure to save your new project on your desktop.

3 Buttons, 3 labels, 1 textbox, and 1 picturebox. Make sure the textbox and the pink label have proper names associated with them (since they will be programmatically manipulated in this lab).

Textbox: TextBoxGuess
Label: LabelFeedback
Buttons: ButtonQuit
ButtonNewGame
ButtonSubmit

Also, make sure your Buttons have appropriate names, so that it is easier to identify them when you are writing code behind each button.  (e.g., ButtonSubmit, ButtonNewGame, and ButtonQuit)

Step 2:  Double click the Quit button and write the code that allows your program to properly exit.

    Close()

Now add the following two lines at the beginning of the program (before the line “Public Class Form1”):

    Option Explicit On
    Option Strict On

Also add the following variable declarations to your program (immediately after the line “Public Class Form1”):

    Dim UserGuess As Integer
    Dim RandomNumber As Integer
    Dim NumberOfGuesses As Integer

Run:  Compile and Run your program.  If your program does not compile, fix the syntax errors and compile the program again.  Once you are able to successfully run the program, check to see if the Quit button works properly.

Step 3:  Save your project and make sure it is saved on you Desktop.  (if you have any doubt about this ask me in the lab)
Step 4: Download the following four images from our OnCource resources and save them on your desktop.

- HighLow.jpg
- Success.jpg
- thumbs-down.jpg
- thumbs-up.jpg

Step 5: Load the picturebox with the “HighLow.jpg” image in to the PictureBox.
Step 6: Double click the **NewGame** button and write the code to initialize your variables. Then use VB’s Random number generator to generate a random number between 0 and 100:

```vba
NumberOfGuesses = 0 ' Starting a new game
UserGuess = 50 ' Give it a default value
LabelFeedback.Text = "Starting a new game, enter your guess above.."

' Generate a random number
Randomize()
RandomNumber = CInt(Rnd() * 100)
```

**Run:** Compile and Run your code again to see if it runs properly. You should notice that when the NewGame button is pressed, the feedback “Starting a new game……” should appear in the label.

Step 7: Double click the **Submit** button and type the following code:

```vba
UserGuess = CInt(TextBoxGuess.Text) 'Take the user's guess and convert it to an integer
If UserGuess < RandomNumber Then
    LabelFeedback.Text = "Higher.."
ElseIf UserGuess > RandomNumber Then
    LabelFeedback.Text = "Lower.."
Else
    LabelFeedback.Text = "You got it !"
End If
```

**Run:** Run your code again to see if it runs properly. Type a number in the textbox, and then click the Submit Button. Follow the directions of the game to see if you can guess the number randomly generated by the program.

Step 8: Once you are confident that the program is running. Try improving the game:

**Question?** How would you improve your game? Can you tell the user how many tries it took them to guess the number? Can you make the program a little more robust? Can you give the user a visual feedback when they guess a wrong number?

**Answer:** Replace the code in step 7 with the code below. That will make your program more robust (see `IsNumeric(TextBoxGuess.Text)`), it keeps track of the number of guesses (see `NumberOfGuesses = NumberOfGuesses + 1` ) and it gives you more visual feedback when the user does the wrong thing!

```vba
If (IsNumeric(TextBoxGuess.Text)) Then
    UserGuess = CInt(TextBoxGuess.Text) 'Take the user's guess and convert it to an integer
    NumberOfGuesses = NumberOfGuesses + 1 'Increment the NumberOfGuesses that user has made
    If UserGuess < RandomNumber Then
        LabelFeedback.Text = "Higher.."
    ElseIf UserGuess > RandomNumber Then
        LabelFeedback.Text = "Lower.."
    Else
        LabelFeedback.Text = "You got it in " & CStr(NumberOfGuesses) & " tries."
    End If
Else
    TextBoxGuess.Text = "" 'clear out the Guess textbox
    TextBoxGuess.BackColor = Color.Red
    MessageBoxIcon.Error
    MessageBox.Show("Enter a number between 1 and 100", _
        "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
    TextBoxGuess.BackColor = Color.White
End If
```
Step 9: Making the game a little more dynamic, by providing more visual feedback to the user. For example, when the user guesses a number that is low than what the program has picked, we would like to display the following image. To indicated that their next guess should be a little higher. To do this we need to have the ability to load a picture into our PictureBox during the execution of the program (while the program is running)

**Loading Images**

In order to load an image in a PictureBox, during the execution of the program, you need the following:

1) Place the pictures that we need in the bin/debug folder of your project.

![Image of bin/debug folder]

2) Add the following Sub Procedure to your code. This sub procedure will load the image into the PictureBox.

   ```vbnet
   Private Sub DisplayImage(ByVal TheImageFile As String)
   Dim MyImage As Bitmap
   PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
   MyImage = New Bitmap(TheImageFile)
   PictureBox1.Image = CType(MyImage, Image)
   End Sub
   
   Private Sub ButtonNewGame_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
   Handles ButtonNewGame.Click
   
   DisplayImage("HighLow.jpg") ' Load the New Game image
   
   End Sub
   ```

3) Now, in order to load different images into the picture box, the programmer must call the above sub procedure. For example at the beginning of the game, we load the “new game” image.
Later when the user guesses a number and we would like to provide visual feedback as to **higher** or **lower** we load different images to provide this feedback.

Add the highlighted lines below to your ButtonSubmit_Click module and run the program to see the result.

```vbnet
Private Sub ButtonSubmit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Handles ButtonSubmit.Click
    If IsNumeric(TextBoxGuess.Text) Then
        UserGuess = CInt(TextBoxGuess.Text) 'Take the user's guess and convert it to an integer
        NumberOfGuesses = NumberOfGuesses + 1 'Increment the NumberOfGuesses that user has made

        If UserGuess < RandomNumber Then
            DisplayImage("thumbs-up.jpg") ' Load the Thumbs Up image
            LabelFeedback.Text = "Higher.."
        ElseIf UserGuess > RandomNumber Then
            DisplayImage("thumbs-down.jpg") ' Load the Thumbs Down image
            LabelFeedback.Text = "Lower..
        Else
            DisplayImage("success.jpg") ' Load the success image
            LabelFeedback.Text = "You got it in " & CStr(NumberOfGuesses) & " tries."
        End If
    Else
        TextBoxGuess.Text = "" 'clear out the Guess textbox
        TextBoxGuess.BackColor = Color.Red
        MessageBox.Show("Enter a number between 1 and 100", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)
        TextBoxGuess.BackColor = Color.White
    End If
End Sub
```