IST 256
Lab Week 8, Day 1 – Monday, October 19, 2009

1. Understanding Procedures

Assume that there is a Form for an application that has one TextBox where the user types in a sales total and one label where the number of units is displayed. There is also the following definition of a general procedure and a button function that calls the procedure:

```
    ' Procedure to compute the number of items from the total
    '  (of a sales) given that the price per unit is $10
    ' and display in a label
    Private Sub ComputeItems(ByRef lb As Label, ByVal total As Single)
        ' Price per unit is fixed at $10
        Dim unitprice as Single
        Dim units as Integer
        unitprice = 10.0
        units = total / unitprice
        lb.Text = "Number of items = " & units
    End Function

    ' User gives the total sales bought in a textbox,
    ' Compute the number of units and display
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim amount as Single
        amount = TextBox1.Text
        Call ComputeItems(Label1, amount)
    End Sub
```

Suppose that the user types the number "100" in the textbox. What will be displayed in Label1.Text? Write your answer here.

Suppose that the user types the number "30" in the textbox. What will be displayed in Label1.Text? Write your answer here.

Suppose that the user types the number "-50" in the textbox. What will be displayed in Label1.Text? Write your answer here.

2. Understanding Procedures with ByRef parameters

Assume that there is a Form for an application that has one TextBox where the user types in a sales total and one label where the amount of tax is shown for that total. There is also the following definition of a general procedure and a button function that calls the procedure:

```
    ' Procedure to compute tax amount from the total (of a sales)
  Private Sub ComputeTax(ByRef tb As TextBox, ByRef taxamount As Single)
        ' Tax rate is fixed at 8%
        Dim taxrate as Single
        taxrate = 0.08
        taxamount = CSng(tb.Text) * taxrate
    End Function

    ' User gives the total sales bought in a textbox,
    ' Compute the amount of tax and Display in a label
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim result as Single

        Call ComputeTax(TextBox1, result)

        ' Show the result
        Label1.Text = "Resulting Tax = " & Format(result, "currency")
    End Sub
```

Suppose that the user types the number "100" in the textbox. What will be displayed in Label1.Text? Write your answer here.

Suppose that the user types the number "3" in the textbox. What will be displayed in Label1.Text? Write your answer here.

3. Testing Predefined Functions

In this lab, you will write a small program to test how some of the predefined functions work. There will be two parts to the program, one to test Math functions and one to test String functions. In the test program, you will test one function at a time, and keep changing the program to test another function.

Open a new project in Visual Studio and call it something like TestFunctions.

Make one set of a Label, TextBox, Button and ResultLabel to test math functions, and another set to test string functions. Your form can look something like:

Type a Number: |_____| |_TestMath_| Result:

Type a String: |_____| |_TestString_| Result:

Math Functions

In the Form program for the TestMath button, you are going to put several calls to math functions from the Predefined Functions sheet, one at a time.

First, declare a variable named number as a Single and assign it the number from the number TextBox.

a. Math.Round:

Add a declaration of a Single variable called rounded and call the function Math.Round with the variable number. Show the result of the function:

Example of the program: (you may need to change textbox and label names)

```
Dim number as Single
Dim rounded as Single
number = CSng(TextBox1.Text)
rounded = Math.Round(number)
Label2.Text = "Result: " & CStr(rounded)
```

Test this with at least three (3) values and record the tests here, by writing the numbers you typed in and the results in the following space. Be sure to test values with decimal points and at least one negative number. Also one of the decimal fractions should be under .5 and one should be over .5.

number	rounded - result of Math.Round(number)

b. Math.Ceiling:
Keep the declaration of number and the statement that assigns to it from the TextBox and remove the other declaration and statement.

Declare an integer variable.

Assign the integer variable to get the result of calling Math.Ceiling(number).

Test this version of the program with at least three values, including one negative number and write the results here:

number	result of Math.Ceiling(number)

c. Choose one of the other Math functions, Math.Sqrt or Math.Abs. Declare a variable to hold the result (what type should it be?), call the function and show the result. Test it with at least three values and write the results here. Be sure to state which function you are testing.

number	result of Math._____(number)

String Functions:
Now we will do a similar set of tests for String functions, using the String TextBox, Button and Result Label.

For all the tests, declare a variable called text of type String and assign it the value of the String TextBox.

a. StrReverse
In the code for the String button, add a declaration for another String variable and add a call to StrReverse and show the result.
Example: (you may need to change textbox and label names)

```
Dim text as String
Dim reversed as String
text = TextBox2.Text
reversed = StrReverse(text)
Label4.Text = "Result: " & reversed
```

Test this program with at least two (2) different strings and write the results of your tests here:

text	result of StrReverse(text)

b. InStr
Keep the code that declares text as a String and assigns it the value of the TextBox.

Declare an integer variable, suppose it is named index
Assign it the result of a call to InStr with the text and another string to find, but just use a single character for the find string. For example:

```
index = InStr(text, "c")
```

Write down which letter you used as a find string and test this on at least three strings and show the results. Be sure to pick example that contains the letter and at least one example that does not.

text	letter	result of Instr(text, letter)

c. Choose another String function from LCase, UCase, Len, and Trim. Declare a variable to get the result of the function (what type should it be?), call the function and show the result. Test with at least three example strings and write the results here. Be sure to state which function you are testing.

text	result of ._____(text)

For this week's lab, hand in the lab page from Monday, the program from Monday and this lab page from Wednesday, and the TestFunctions program as it ended up.