Lab 6: Repetition Logic

Visual Basic.NET allows a procedure to be repeated many times as long as the processor and memory could support. This is generally called looping. Looping is required when we need to process something repetitively until a certain condition is met. For example, we can design a program that adds a series of numbers until the sum exceeds a certain value, or a program that asks the user to enter data repeatedly until he/she keys in the word 'Finish'. In Visual Basic.NET, we have three types of Loops, they are the For…..Next loop, the Do loop and the While…..End while loop.

1. Conditional Operators

Syntax	Example
' for-loop For counter=startNumber to endNumber Step increment body Next	' Sum from 1 to 100 For count = 1 to 100 sum += number Next
' do while loop Do while/ While (test) body Loop/ End While	Dim number As Integer = 1 Do while number <= 100 sum += number number+=1 Loop
' do loop-while loop Do body Loop while (test)	Dim number As Integer = 1 Do sum += number number+=1 Loop while number <= 100
' do until loop Do Until (test) body Loop	Dim number As Integer = 1 Do until number > 100 sum += number number+=1 Loop
' do loop-until loop Do body Loop Until (test)	Dim number As Integer = 1 Do sum += number number+=1 Loop until number > 100

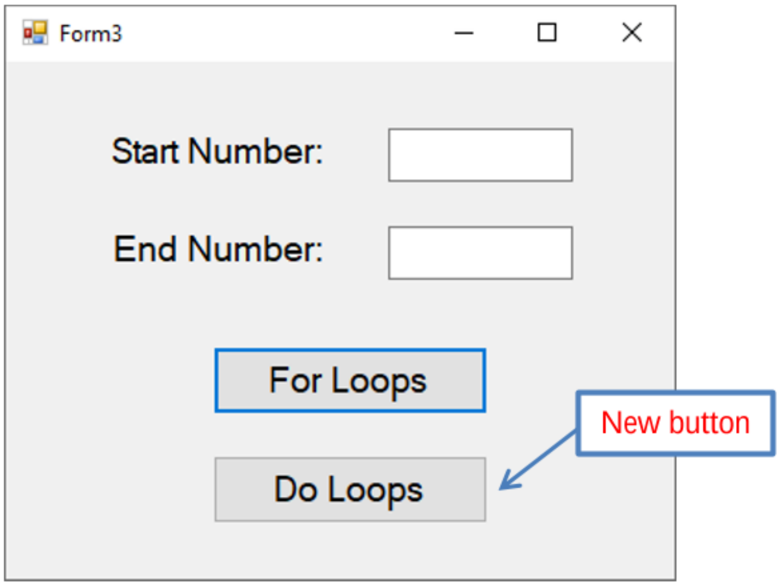
Exercise 1

Put two textboxes on your form. The first box asks users to enter a start position for a For Loop; the second textbox asks user to enter an end position for the For loop. When a button is clicked, the program will add up the numbers between the start position and the end position. Display the answer in a message box. You can use this For Loop code:

```
Dim answer as Integer = 0
For i = startNumber To endNumber
        answer = answer + i
Next
MsgBox (answer)
```

You need to get the startNumber and endNumber value from the textboxes. When you've finished, run the program and see what happens.

Add another button to the Form. Your form might look something like this:



Double click the new button to open the code window, and then type the following code for the new button:

```
Dim number as Integer
number = 1
Do While number < 5
        MsgBox(number)
        number = number + 1
Loop
```

When you've finished, run the program and see what happens. The numbers 1 to 4 should have displayed in your message box.

So the Do loop keeps going round and around. And when it gets to the top, it tests the "While" part - Do While number is Less Than 5. It's really answering a Yes or No question: is the number inside the variable called number Less Than 5? If it is Less Than 5, go round the loop again. If it's not Less than 5, Visual Basic jumps out of the Loop entirely.

You can add the "While ... " part to the bottom, just after the word "Loop". Like this:

```
Dim number as Integer
number = 1
Do
    MsgBox(number)
    number = number + 1
Loop While number < 5
```

Try it and see what difference it makes.

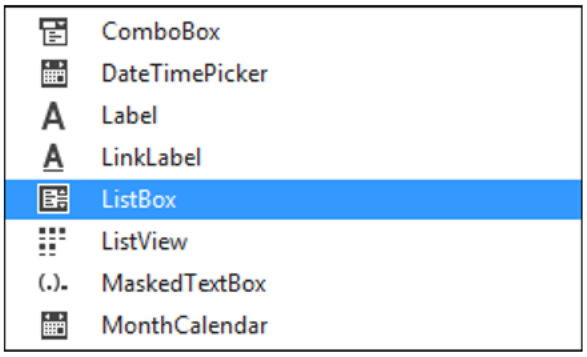
Exercise 2 - Times Table Program

Start a new project for this. Onto your new Form, place two textboxes and a button. Set the Text property of Textbox1 to 1, and the Text property of Textbox2 to 10. Set the Text property of the Button to "Go".

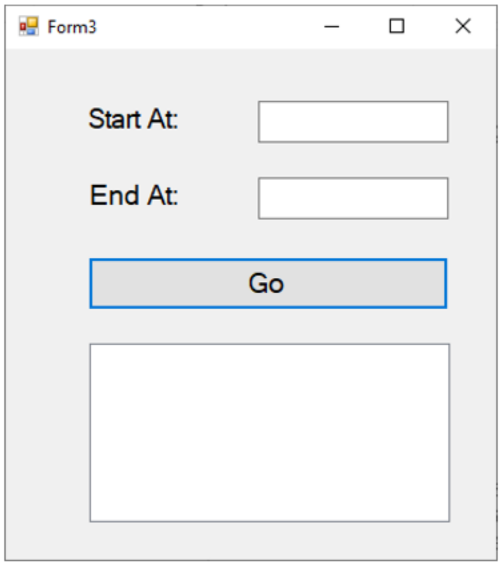
When the Go button is clicked, the program will put the numbers from the Textbox into two variables. We'll then put a value into a variable called multiplier. If you're doing the times tables, the format is

X multiplied by Y = Z
(2 multiplied by 3 = 6)

We'll use a For Loop to work out the X part; we'll get the Y part from a multiplier variable. We'll then display the results in something called a Listbox. So add a List Box to your form. It looks like this in the toolbox:



The form you design should look something like this one:



A List box is similar to a Combo Box, in that you have a list of items that the user can select. Here, we're just using it display the results of our program. We'll add items to the List box with our code, rather than in design time like we did for the Combo box.

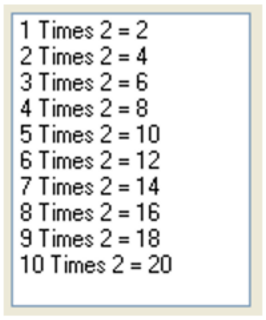
So, here's the code for the entire program. Double click your Go button, and add the following:

```
Dim number1 As Integer
Dim number2 As Integer
Dim multiplier As Integer
Dim answer As Integer
Dim i As Integer

number1 = Val(TextBox1.Text)
number2 = Val(TextBox2.Text)
multiplier = 2

For i = number1 To number2
        answer = i * multiplier
        lstResult.Items.Add(i & " Times " & multiplier & " = " & answer)
Next
```

When you've finished, run the program and see how it works. You should see this appear in your List box:



```
1 Times 2 = 2
2 Times 2 = 4
3 Times 2 = 6
4 Times 2 = 8
5 Times 2 = 10
6 Times 2 = 12
7 Times 2 = 14
8 Times 2 = 16
9 Times 2 = 18
10 Times 2 = 20
```

As you can see from the code, we've set up five Integer variables - number1, number2, multiplier, answer and i. The next thing we did is to pass whatever is in our two Textboxes straight into the two variables, number1 and number2. The start number goes into textbox one, and the end number goes into textbox2.

```
number1 = Val(TextBox1.Text)
number2 = Val(TextBox2.Text)
```

In the next line of code, we set a starting value for the multiplier variable:

```
multiplier = 2
```

The next thing we had was a For Loop. The For Loop was this:

```
For i = number1 To number2
    answer = i * multiplier
    lstResult.Items.Add(i & " Times " & multiplier & " = " & answer)
Next
```

Remember: the number1 and number2 variables hold our numbers from the Textboxes. We set these to 1 and 10. So our first line of the For Loop is really this:

```
For i = 1 To 10
```

We're saying, "Start a For Loop". Whatever is in the variable called number1, make that the starting number for the Loop. Put this value into the variable called i. The end of the Loop will come when the variable called i has the value 10. Stop looping when you reach this value.

The next part of the code reads this:

```
answer = i * multiplier
```

This means, put into the variable called answer the following sum: whatever is in the variable called i multiplied by whatever is in the variable called multiplier.

Finally, a word about the line that displays your text in the list box. It was this:

```
lstResult.Items.Add(i & " Times " & multiplier & " = " & answer)
```

To add items to a list box with code, first you type the name of your list box:

```
lstResult
```

Type a full stop and a drop down list will appear. Select Items from the list.

```
lstResult.Items
```

Type another full stop and again a drop down list will appear. Select the Add Method

```
lstResult.Items.Add
```

This method, not surprisingly, lets you add items to your list box. Whatever you want to add goes between a pair of round brackets:

```
lstResult.Items.Add( )
```

In between the round brackets, we have this for our code:

```
i & " Times " & multiplier & " = " & answer
```

It might be a bit long, but there are 5 parts to it, all joined together by the concatenate symbol (&):

```
i
" Times "
multiplier
" = "
answer
```

The variable i holds the current value of the For Loop; " Times " is just direct text; multiplier holds the value we're multiplying by (our times table); " = " is again direct text; and answer is the answer to our times table sum.

If you want to clear the items from a List box you can do this. At the top of the code, enter this line:

```
ListBox1.Items.Clear()
```

Add another textbox to your form. This will represent the "times table". So far you have been getting this value directly from the code. For the last exercise, get the multiplier value directly from the textbox. Add appropriate labels to all your textboxes.

Exercise 1

In exercise 1.1 through 1.3, determine the output displayed when the button is clicked on.

1.1
```
Private Sub btnDisplay_Click(…) Handles btnDisplay.Click
Dim num As Integer = 1
Dim total As Integer = 0
Do
    total = total + num
    num = num + 1
Loop Until num >= 5
txtOutput.Text = total
End Sub
```

1.2
```
Private Sub btnDisplay_Click(…) Handles btnDisplay.Click
Dim num As Integer = 3
Do While num < 15
    num = num + 5
Loop
txtOutput.Text = num
End Sub
```

1.3
```
Private Sub btnDisplay_Click(…) Handles btnDisplay.Click
For i = 1 To 4
    lstBox.Items.Add("Pass #" & i)
Next
End Sub
```

Exercise 2

Write a program containing a loop to find the value of 1 + 2 + 3 + … + 100.

Exercise 3

Write a program that display a Celsius-to-Fahrenheit conversion table in a list box. Entries in the table should range from 0 to 100 degree Celsius in increment of 10 degrees. Note: The formula Fahrenheit = (9/5 * Celsius) + 32 converts Celsius to Fahrenheit.

Exercise 4 Genius question

Create a program to estimate how much a young worker will make before retiring at age 65. Request the worker's name, age, and starting salary as input. Assume the worker received a 5% raise each year. For example, if the user enters Helen, 25, 20000, then the text box should display the following:

Helen will earn about RM2415995.

Exercise 5 Genius question

Create a program called HarmonicSum to compute the sum of a harmonic series, as shown below, where n is the user input value. The program shall compute the sum from left-to-right. Beware that int/int gives int.

$$Harmonic(n) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$