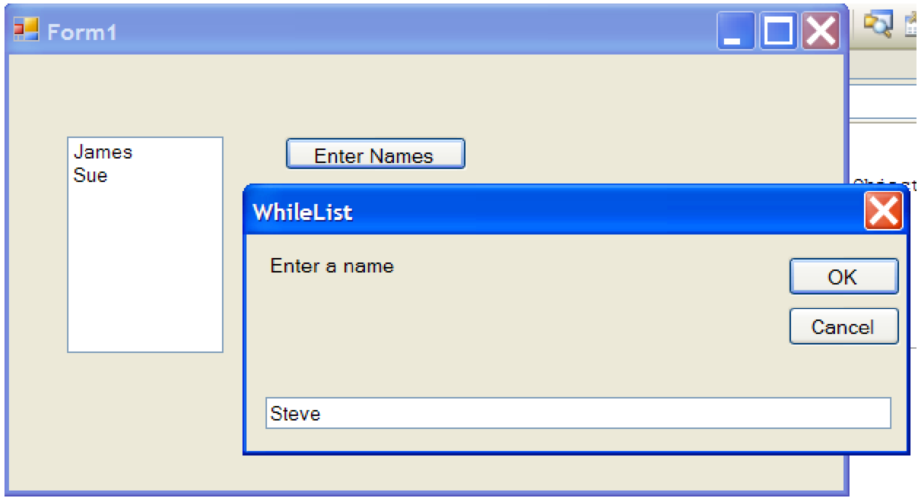
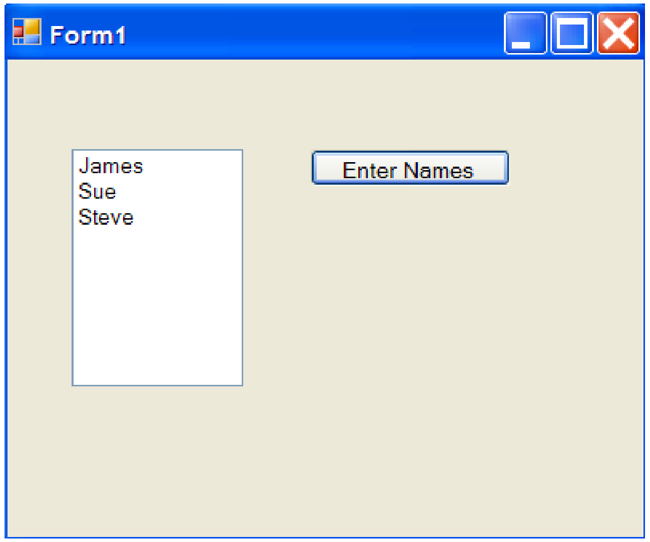
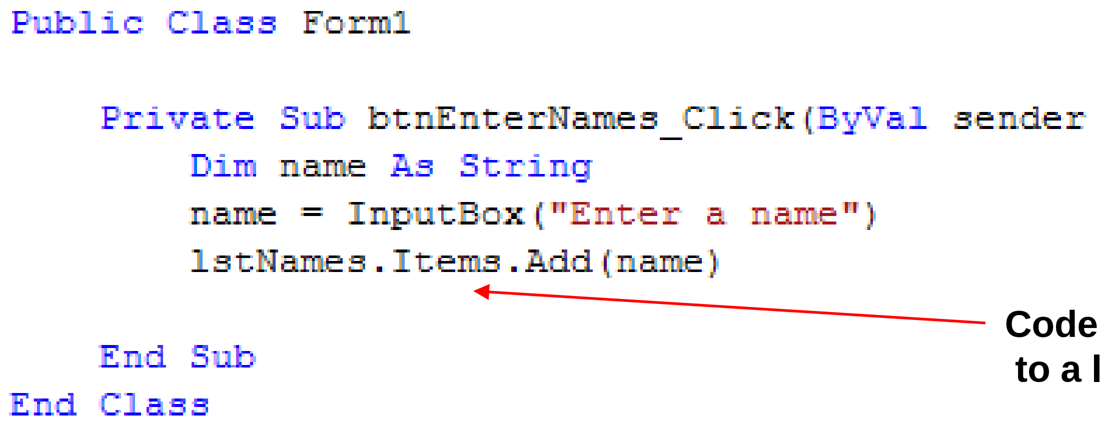
The List Box

In this example we will create a list of names that will appear in a list box. Using an Input Box we can prompt the user for a name. In this first example we will require the user to keep clicking the Enter Name button each time they want enter another name.





List Box Code



```
Public Class Form1

    Private Sub btnEnterNames_Click(ByVal sender
        Dim name As String
        name = InputBox("Enter a name")
        lstNames.Items.Add(name)

    End Sub
End Class
```

Code for adding an item (name variable) to a list box

Looping the Name Entry

Lets modify the program so that the user no longer has to click ‘Enter Name’ each time they want to enter an item.

Instead the program will continuously prompt the user for another name until they type in ‘q’.

```
Private Sub btnEnterNames_Click(ByVal sender As System.Object

    Dim name As String
    name = InputBox("Enter a name")
    While (name <> "q")
        lstNames.Items.Add(name)
        name = InputBox("Enter a name")
    End While


End Sub
```

User must be prompted for input before they enter the loop…

and

again, within the loop.
Typing in ‘q’ at any time will cause the program to (not enter/exit) the loop.

The ***For-Next*** Loop

While loops are useful if it is not known ahead of time how many times the loop will need to repeat. In many cases the number of repetitions of the loop are already known. In cases like this, a ***for-next loop*** can be used (This is more commonly known as a ***for loop***).

Imagine wanting to know the sum of numbers from 1 to 10.

1+2+3+4+5+6+7+8+9+10

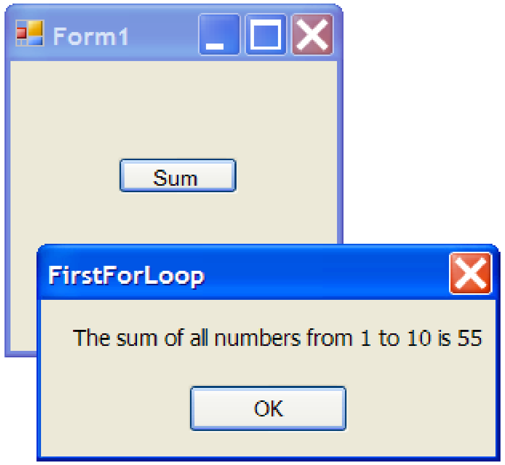
You already know that the loop will need to repeat 10 times.

Lets use the variable count to keep track of the repetitions and tally to keep track of the sum.

For Loop

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As

    Dim count As Integer
    Dim tally As Integer = 0

    For count = 1 To 10

        tally = tally + count

    Next count

    MsgBox("The sum of all numbers from 1 to 10 i

End Sub
```

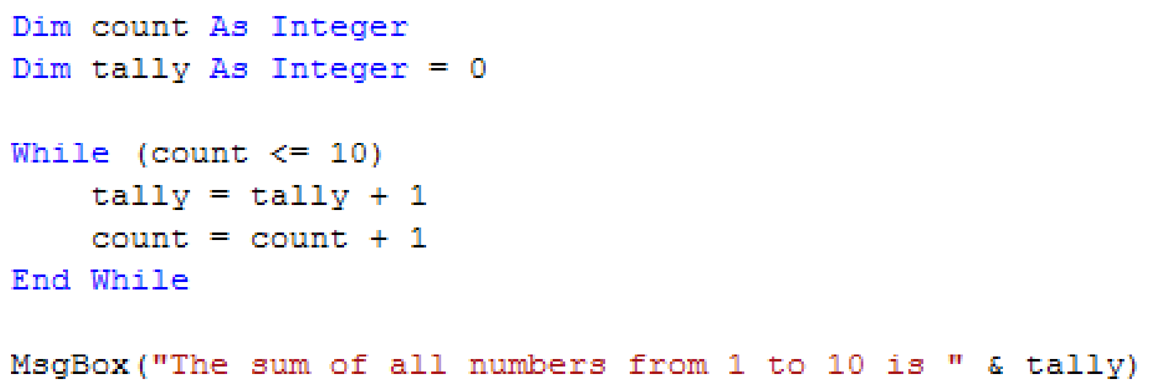
This tells us that the loop will repeat 10 times and that the value of count will increment by one each time.



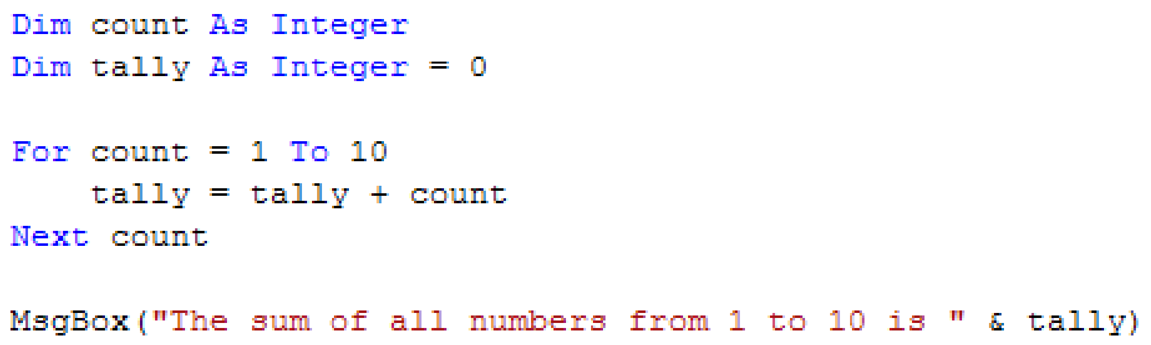
The sum of all numbers from 1 to 10 is 55

Comparing ***For Loops*** and ***While Loops***

Here is the same program written using both a ***while loop*** and a ***for loop***.



```
Dim count As Integer
Dim tally As Integer = 0

While (count <= 10)
    tally = tally + 1
    count = count + 1
End While

MsgBox("The sum of all numbers from 1 to 10 is " & tally)
```



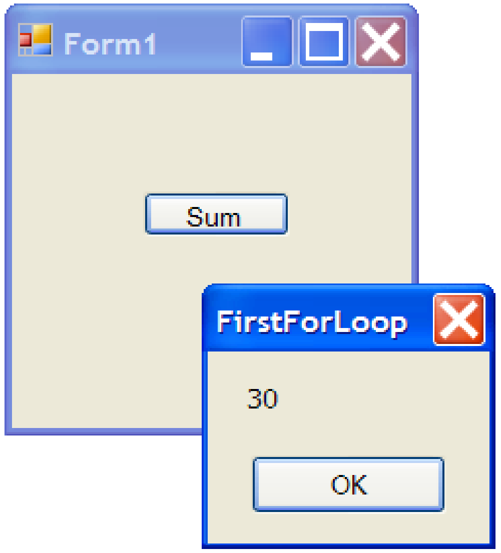
```
Dim count As Integer
Dim tally As Integer = 0

For count = 1 To 10
    tally = tally + count
Next count

MsgBox("The sum of all numbers from 1 to 10 is " & tally)
```

Step

Using while loops we were able to increment the counter variables by values other than 1.

eg .

```
While (count <= 10)
    tally = tally + count
    count = count + 2
End While
MsgBox(tally)
```



30

```
Dim count As Integer
Dim tally As Integer = 0

For count = 1 To 10 Step 2
    tally = tally + count
Next count

MsgBox("The sum of every other number from 1 to 10 is " & tally)
```

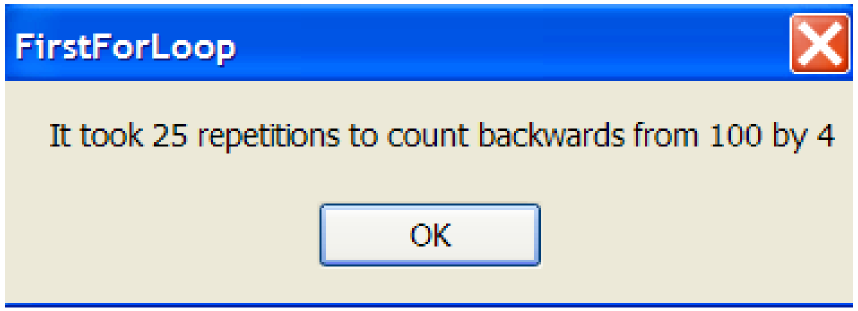
Using Step 2 in a For Loop has the same effect

Counting Backwards

For Loops are flexible in that they can be used to count backwards by one or some other step.

Backwards by -4

```
Dim count As Integer = 100
Dim tally As Integer = 0

For count = 100 To 1 Step -4
    tally = tally + 1
Next count

MsgBox("It took " & tally & " repetitions to count backwards from 100 by 4")
```



It took 25 repetitions to count backwards from 100 by 4