The For – Next Statement

Loops are used to repeat a section of code again and again. The **For – Next** statement is one of the principal looping statements of Basic.

A **For – Next** loop is called a *definite* loop. In a definite loop, the number of times the code contained in the loop will repeat is generally known when the loop begins.

For *CounterVariable* = *StartValue* to *EndValue* [Step *Increment*]
          Statement
          (more statements may follow)
Next [*CounterVariable*]

CounterVariable – is the variable to be used as the counter. It must be a number.
StartValue is the value the counter variable will be initially set to.
EndValue is the value the counter variable is tested against just prior to each iteration of the loop.
Step Increment - is the amount added to the counter variable at the end of each iteration. The default is 1.
Next[CounterVariable] statement marks the end of the loop and causes the counter variable to be incremented.

1.
```
Dim intCount as Integer
For intCount = 1 to 10
  lstOutput.Items.Add intCount
Next intCount
```

2.
```
Dim intCount as Integer
For intCount = 0 to 10 Step 2
lstOutput.Items.Add intCount
Next intCount
```

3.
```
Dim x as Integer
For x = 100 to 0 Step -5
  lstOutput.Items.Add x
Next x
```

4.
```
Dim x as Integer
For x = 1 to 5
lstOutput.Items.Add “Hello”
Next x
```

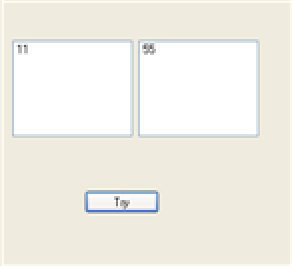
The **Tab(n)** function may be included to skip to print zones in the output. The print zones are equivalent to the tab settings in a text document.

5.
```
Dim x as Integer, FirstName as String, LastName as String, Address as String
FirstName = “Bart”
LastName = “Simpson”
Address = “Miramichi”
For x = 1 to 5
      lstOutput.Items.Add(FirstName)
      lstOutput.Items.Add(LastName)
      lstOutput.Items.Add(Address)
Next x
```

Running Totals

An assignment statement can be placed inside a loop that will keep a running total of certain values. Think of all the times information must be totaled in computer programs.
Use two different listboxes.
eg.

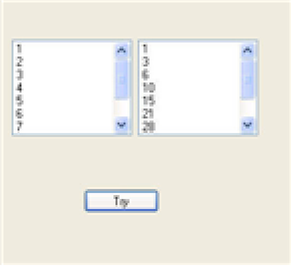
```
6.    Dim x As Integer, Total As Integer
        For x = 1 To 10
            Total = Total + x
            lstx.Items.Add(x)
            lstOutput.Items.Add(Total)
              Next x
```



```
11
```

```
55
```

```
6b.   Dim x As Integer, Total As Integer
        For x = 1 To 10
            Total = Total + x
               Next x
             lstx.Items.Add(x)
            lstOutput.Items.Add(Total)
            Next x
```



```
1
2
3
4
5
6
7
```

```
1
3
6
10
15
21
28
```

Listboxes

We could also display the results in a **listbox**. To add items to a listbox, you use the **AddItem** method. Listboxes will automatically add a scrollbar if needed to view the display. Add a listbox named NameList. The **&** character is used to join separate strings to make one long string.

```
7.     Dim Name As String, TbCh As String
         Dim x As Integer, FirstName As String, LastName As String, Address As String
         FirstName = "Homer"
         LastName = "Simpson"
         Address = "Miramichi"
         TbCh = "    "
         For x = 1 To 15
             Name = FirstName & TbCh & LastName & TbCh & Address
             lstName.Items.Add(Name)
         Next x
```

The **Clear** method can be used to clear or erase all items from a listbox.
eg.   NameList.Clear

If you need to have two variables to count at the same time, you could use the For – Next method for one and the **x=x+1** method for the other. eg.

```
6.      Dim x as Integer, Y as Integer
        x = 0
        For y = 1 to 3
                x = x+2
                picDisplay.Print y; Tab(10); x
        Next y
```

```
7.      Dim x as Integer, Y as Integer
        y=5
        For x = 0 to 10 step 2
                y = y + 1
                picDisplay.Print x; Spc(5); y
        Next x
```

Breaking out of a Loop

Although the **For** line sets the number of times the program will loop, it is possible to break out of the loop depending on if a certain condition is met. eg.

```
8.      Dim x as Integer, y as Integer
        For x = 1 to 10
                y = x * 2
                If y = 8 then Exit For
                picDisplay.Print x; tab(7); y
        Next x
```

Nested Loops

You can have a loop within a loop. If so, one must be completely inside the other. In other words, the one that starts first must end last. Notice that the loops are indented so that you can quickly see what is inside the loop. eg.

```
9.   Dim x as Integer, y as Integer
        For x = 1 to 3
                For y = 1 to 5
                        picDisplay.Print x; tab(10); y
                Next y
        Next x
```

Running Totals

An assignment statement can be placed inside a loop that will keep a running total of certain values. Think of all the times information must be totalled in computer programs.
eg.

```
10.     Dim x as Integer, Total as Integer
        For x = 1 to 10
                Total = Total + x
        Next x
        picDisplay.Print Total
```

Conversion Tables

For – Next Loops provide a handy means of converting values, such as Imperial and Metric measurements, and printing out a conversion table. eg.

11.

```
Dim Inches as Single, Centimetres as Single
picDisplay.Print “Centimetres”; tab(15); “Inches”
For Centimetres  = 1  to 10
        Inches = Centimetres / 2.54
        picDisplay.Print Centimetres; tab(15);  Inches
Next Centimetres
```

Listboxes

We could also display the results in a **listbox**.   To add items to a listbox, you use the **AddItem** method. Listboxes will automatically add a scrollbar if needed to view the display. Add a listbox named NameList. The **&** character is used to join separate strings to make one long string.

12.

```
Dim Name as String, TbCh as String
Dim x as Integer, FirstName as String, LastName as String, Address as String
FirstName = “Marg”
LastName = “Barker”
Address = “Miramichi”
TbCh = Chr$(9)
For x = 1 to 15
        Name = FirstName & TbCh & LastName & TbCh & Address
        lstName.AddItem Name
Next x
```

The **Clear** method can be used to clear or erase all items from a listbox.

eg.

```
NameList.Clear
```

PROGRAMMING PROBLEMS

1. Create a program with a picturebox in which to display information. Add a command button which would do each of the following (Make the required changes, one by one.):

   (a) count by 1’s from 0 to 5
   (b) count by 1’s from 1 to 12
   (c) count by 2’s to show the even numbers from 0 to 20
   (d) count by 2’s to show the odd numbers from 1 to 15
   (e) count backwards from 10 to 1
   (f) display your first name 10 times and have the lines numbered

2. Create a program to display a table under the headings KMH and MPH. Kilometres should be listed under the appropriate heading from 40 to 120 with the equivalent conversions under MPH. Do this in steps of 10. 1km = .6213 miles

3. Create a program to **add** all the numbers from 1 to 20 and display only the answer. Then modify the program to add all the even numbers from 0 to 50.

4. Create a program that will convert fahrenheit degrees to celsius. The range of fahrenheit degrees is to go from 32 to 212 in steps of 10. Print out a table showing corresponding values under the headings "Fahrenheit" and "Celsius." To calculate the number of degrees celsius, first subtract 32 from the number of F degrees and then take 5/9 of the remainder.

5. Given ($Y=3X^3-4X^2+X-1$), have a table of values printed out. Under X, have the values from –20 to 20 in steps of 2 shown. Under Y have the corresponding values. Put this display in a listbox.

6. Create a program to display:
   (a) the five times table
   (b) all the times tables, one after another

7. Create an Average application that calculates the average of a series of numbers from 1 to a number entered by the user. For example, if the user enters 5 then the application would calculate the average of 1, 2, 3, 4, and 5.

8. Create a Sum application that calculates the sum of the numbers between two numbers inclusive entered by the user. For example, if the user enters 10 and 15, then the application would calculate 10+11+12+13+14+15.

9. Create a conversion table of X and Y values that begins with X = 0 and ends when Y>20, given the following equation: $Y = X^2 – X + 2$

10. Suppose you decide to save your pennies. The first day you save one penny, the second day twice as much as the day before; that is 2 cents; the next day twice the previous day; i.e. 4 cents; and so on for thirty days. Guess how much you would have saved, then write a program to calculate it. Show this in table form under the following headings:
    DAY NO. SAVED TOTAL SAVED

11. Create a program to print a table of values on the form when you hit a command button. Mr. Jones borrows $6000 from the credit union to trade in his car. He will repay the loan at the rate of $400 per month plus interest of 2% on the unpaid balance. Your program will print out a schedule of payments and calculate the total amount repaid. Make up a table under the headings:

	UNPAID	INTEREST ON		TOTAL
MONTH	BALANCE	PRINCIPAL	PAYMENT	PAYMENT
1	6000	120	400	520
2	5600	112	400	512

Sample Programs—What will be displayed by each?

1.

```
Dim x as Single
For x = 1 to 8 step 3
    picDisplay.Print x
Next x
```

2.

```
dim x as Single, y as Single
For x = 1 to 5
    y = y + x
    picDisplay.Print x; tab(15);y
Next x
```

3.

```
dim x as Single, y as Single
For x = 10 to 0 step –2
    y = x * 2
    picDisplay.Print y
Next x
```

4.

```
dim x as Integer
For x = 1 to 5
Next x
picDisplay.Print x
```

5.

```
dim x as Integer
For x = 3 to 6
    picDisplay.Print “Hi there”
Next x
```

6.

```
dim x as Integer, y as Single
For x = 1 to 4
    y = y + 2
    picDisplay.Print x; tab(7);y
Next x
```

7.

```
dim x as Single, y as Single, FirstName as String
FirstName = “Alex”
For x = 1 to 5
    y = x
    picDisplay.Print y; tab(5); FirstName
Next x
```