Programming Logic & Design

Chapter 5 – Repetition Structures (Part 2)

Queen's College CSD

1133 – CPCM -2023S

Chapter 5 – Repition Structures

Topics:

- Count-Controlled Loops and the For Statement.
- Calculating a Running Total
- Sentinels
- Nested Loops

Chapter 5 – Repeition Structures

Count-Controlled Loops and the For Statement

Programmers commonly have to write code that performs the same task over

A count-controlled loop iterates a specific number of times. Although you can write the logic of a condition-controlled loop so it iterates a specific number of times, most languages provide a loop known as the For loop, which is specifically designed as a count-controlled loop.

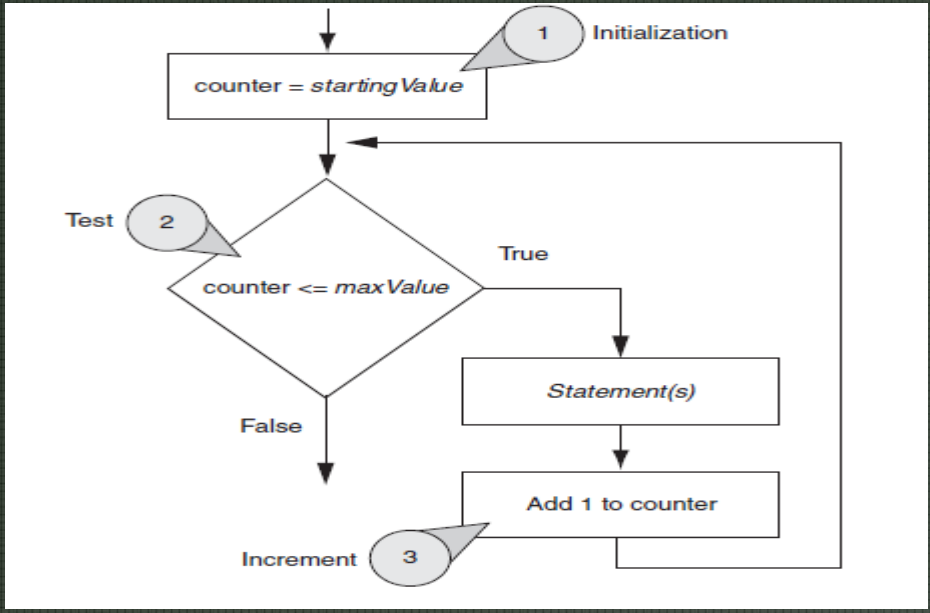
The way that a count-controlled loop works is simple: the loop keeps a count of the number of times that it iterates, and when the count reaches a specified amount, the loop stops. A count-controlled loop uses a variable known as a counter variable, or simply counter, to store the number of iterations that it has performed. Using the counter variable, the loop typically performs the following three actions: initialization, test, and increment:

1. **Initialization**: Before the loop begins, the counter variable is initialized to a starting value. The starting value that is used will depend on the situation.

Chapter 5 – Repeition Structures

2.Test: The loop tests the counter variable by comparing it to a maximum value. If the counter variable is less than or equal to the maximum value, the loop iterates. If the counter is greater than the maximum value, the program exits the loop.

3.Increment: To increment a variable means to increase its value. During each iteration, the loop increments the counter variable by adding 1 to it.



Chapter 5 – Repeition Structures

The For Statement

Count-controlled loops are so common in programming that most languages provide a statement just for them. This is usually called the For statement. The For statement is specifically designed to initialize, test, and increment a counter variable

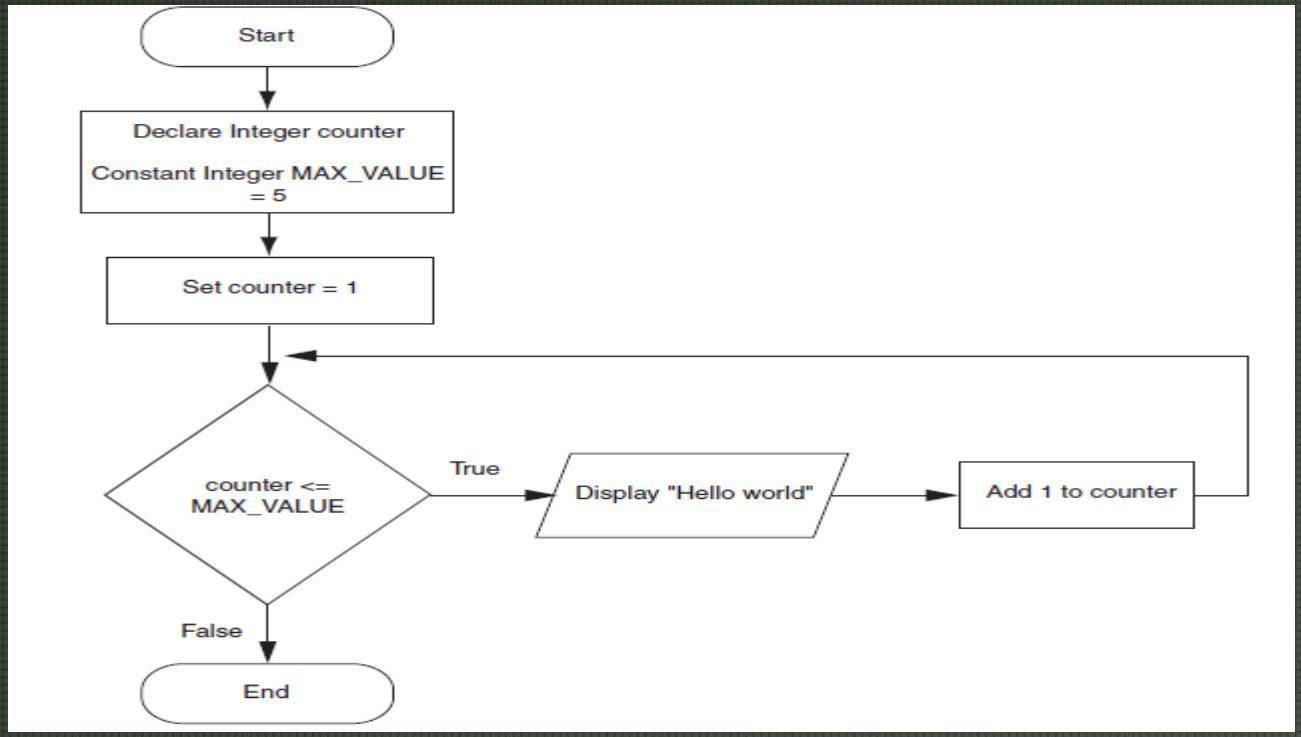
```
For counterVariable = startingValue To maxValue
    statement
    statement
    statement
    etc.
End For
```

These statements are the body of the loop.

```
1 Declare Integer counter
2 Constant Integer MAX_VALUE = 5
3
4 For counter = 1 To MAX_VALUE
5    Display "Hello world"
6 End For
```

Notice that the loop does not contain a statement to increment the counter variable. This happens automatically in a For loop, at the end of each iteration. For that reason, you should be careful not to place a statement that modifies the counter variable inside the body of a For loop. Doing so will usually disrupt the way the For loop works.

Chapter 5 – Repeition Structures



Example:

write a program that displays the numbers 1 through 10 and their squares, in a tabular format:

Chapter 5 – Repeition Structures

```
1  // Variables
2  Declare Integer counter, square
3
4  // Constant for the maximum value
5  Constant Integer MAX_VALUE = 10
6
7  // Display table headings.
8  Display "Number", Tab, "Square"
9  Display "----------------------"
10
11 // Display the numbers 1 through 10 and
12 // their squares.
13 For counter = 1 To MAX_VALUE
14    // Calculate number squared.
15    Set square = counter^2
16
17    // Display number and number squared.
18    Display counter, Tab, square
19 End For
```

Program Output

```
Number          Square
----------------------
     1               1
     2               4
     3               9
     4              16
     5              25
     6              36
     7              49
     8              64
     9              81
    10             100
```

Task:

Write the flowchart for the above program

Explain Answer in class5Examples

Chapter 5 – Repeition Structures

Incrementing by Values Other Than 1

The amount by which the counter variable is incremented in a For loop is known as the step amount. By default, the step amount is 1. Most languages provide a way to change the step amount. This gives you the ability to increment the counter variable by any value you wish.

In pseudocode, we will use the Step clause to specify a step value in a For loop.

```
For counter = 0 To 100 Step 10
   Display counter
End For
```

Another example

```
1  // Declare a counter variable
2  Declare Integer counter
3
4  // Constant for the maximum value
5  Constant Integer MAX_VALUE = 11
6
7  // Display the odd numbers from 1 through 11.
8  For counter = 1 To MAX_VALUE Step 2
9     Display counter
0  End For
```

Program Output

```
1
3
5
7
9
11
```

Chapter 5 – Repeition Structures

Problem :

Your friend Amanda just inherited a European sports car from her uncle. Amanda lives in the United States, and she is afraid she will get a speeding ticket because the car's speedometer works in kilometers per hour. She has asked you to write a program that displays a table of speeds in kilometers per hour with their values converted to miles per hour. The formula for converting kilometers per hour to miles per hour is:

$$MPH = KPH \times 0.6214$$

In the formula, *MPH* is the speed in miles per hour and *KPH* is the speed in kilometers per hour.

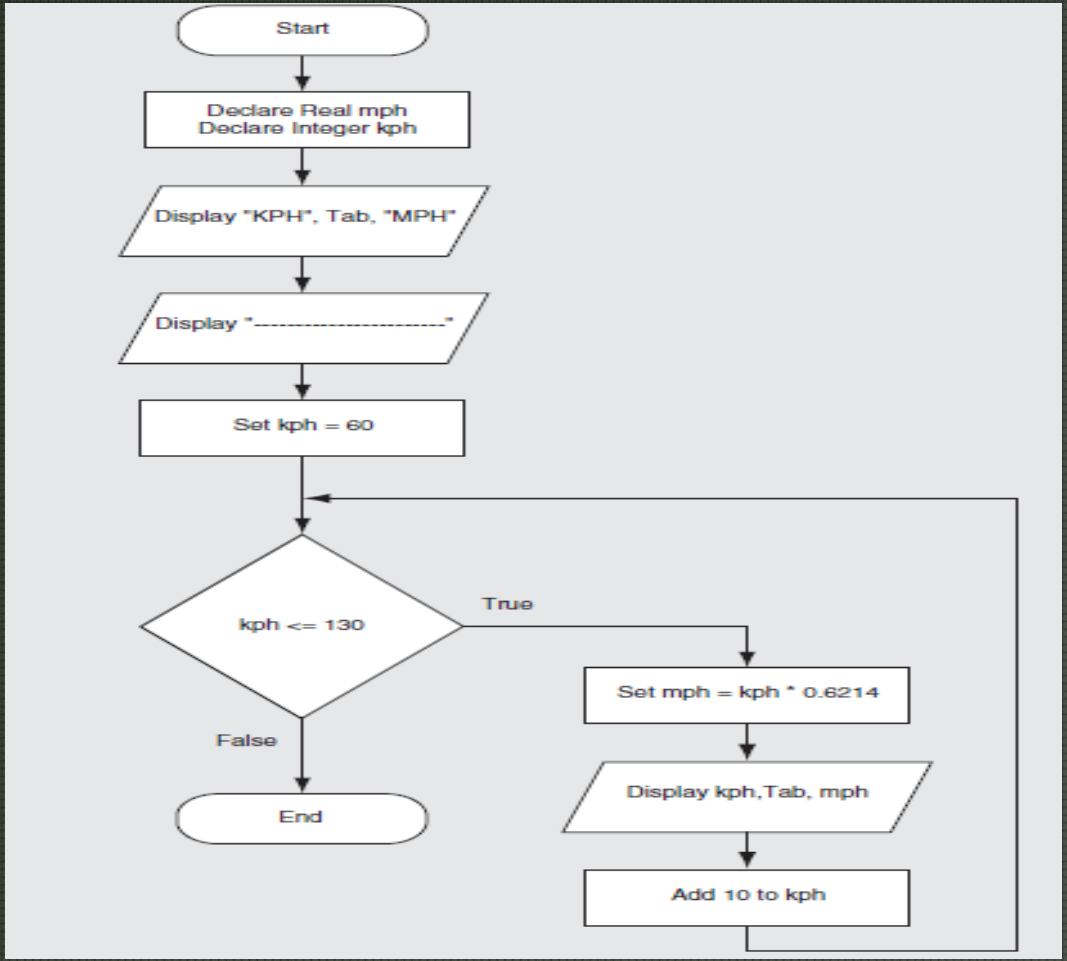
The table that your program displays should show speeds from 60 kilometers per hour through 130 kilometers per hour, in increments of 10, along with their values converted to miles per hour. The table should look something like this:

KPH	MPH
60	37.284
70	43.498
80	49.712
etc. . . .	
130	80.782

Chapter 5 – Repeition Structures

Solution:

Flowchart given below try to design the pseudocode this time



Explain Answer in class5Examples

Chapter 5 – Repeition Structures

Counting Backward by Decrementing the Counter Variable:

Although the counter variable is usually incremented in a count-controlled loop, you can alternatively decrement the counter variable. To decrement a variable means to decrease its value. In a For statement, you specify a negative step value to decrement the counter variable. For example, look at the following loop:

```
For counter = 10 To 1 Step -1
Display counter
End For
```

Task:

Sometimes, however, the programmer needs to let the user decide the number of times that a loop should iterate. For example, what if you want Program 5-9 to be a bit more versatile by allowing the user to specify the maximum value displayed by the loop? Design the pseudocode to accomplish this.

Explain Answer in class5Examples

Chapter 5 – Repeition Structures

Designing a Count-Controlled While Loop

In most situations, it is best to use the For statement to write a count-controlled loop. Most languages, however, make it possible to use any looping mechanism to create a count-controlled loop. For example, you can create a count-controlled While loop, a count-controlled Do-While loop, or a count-controlled Do-Until loop. Regardless of the type of mechanism that you use, all count-controlled loops perform an initialization, test, and increment operation on a counter variable.

In pseudocode, you can use the following general format to write a count-controlled `While` loop:

```
① Declare Integer counter = startingValue      ← Initialize a counter variable to the starting value.
② While counter <= maxValue                    ← Compare the counter to the maximum value.
      statement
      statement
      statement
③     Set counter = counter + 1                ← Add 1 to the counter variable during each iteration.
End While
```

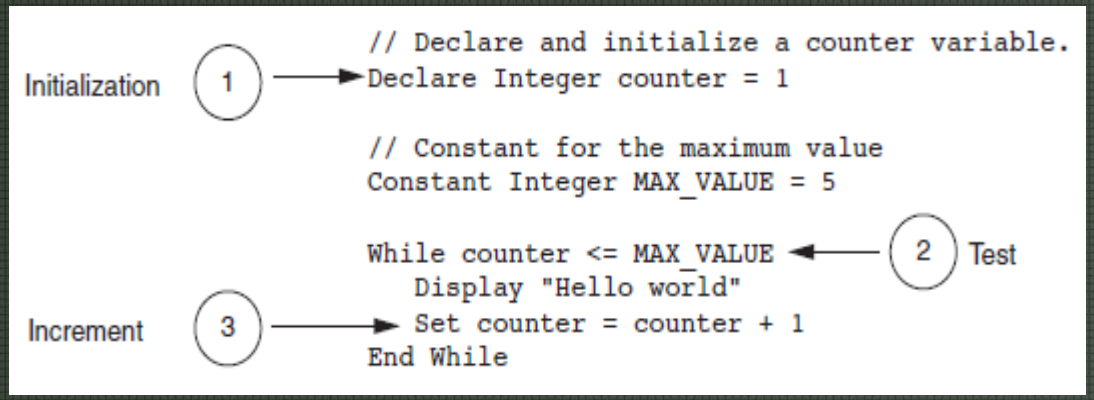
Chapter 5 – Repeition Structures

The pseudocode shown below is an example of a count-controlled While loop. This program follows the same logic that we discussed previously (it displays “Hello world” five times.

```
1  // Declare and initialize a counter variable.
2  Declare Integer counter = 1
3
4  // Constant for the maximum value
5  Constant Integer MAX_VALUE = 5
6
7  While counter <= MAX_VALUE
8     Display "Hello world"
9     Set counter = counter + 1
10 End While
```

Program Output

```
Hello world
Hello world
Hello world
Hello world
Hello world
```



Task : Design a pseudocode for the problem in slide 7 using count-controlled while loop Explain Answer in class5Examples

Chapter 5 – Repeition Structures

Checkpoint Questions

What is a loop iteration?

What is the difference between a pretest loop and a posttest loop?

Does the While loop test its condition before or after it performs an iteration?

Does the Do-While loop test its condition before or after it performs an iteration?

What is an infinite loop?

What is the difference between a Do-While loop and a Do-Until loop?

What is a counter variable?

What three actions do count-controlled loops typically perform using the counter variable?

When you increment a variable, what are you doing? When you decrement a variable, what are you doing?

Look at the following pseudocode. If it were a real program, what would it display?

```
Declare Integer number = 5
Set number = number + 1
Display number
```

Chapter 5 – Repeition Structures

Sentinels

CONCEPT: A sentinel is a special value that marks the end of a list of values.

Consider the following scenario:

You are designing a program that will use a loop to process a long list of values. At the time you are designing the program, you do not know the number of values that will be in the list. In fact, the number of values in the list could be different each time the program is executed. What is the best way to design such a loop? Here are some techniques that you have seen already in this chapter, along with the disadvantages of using them when processing a long list of values:

- Simply ask the user, at the end of each loop iteration, whether there is another value to process. If the list of values is long, however, asking this question at the end of each loop iteration might make the program cumbersome for the user.
- Ask the user at the beginning of the program how many items the list contains. This might also inconvenience the user, however. If the list is very long, and the user does not know the number of items in the list, it will require the user to count them.

Chapter 5 – Repeition Structures

Sentinels

Problem:

Using a Sentinel

The county tax office calculates the annual taxes on property using the following formula:

$$Property\ Tax = Property\ Value \times 0.0065$$

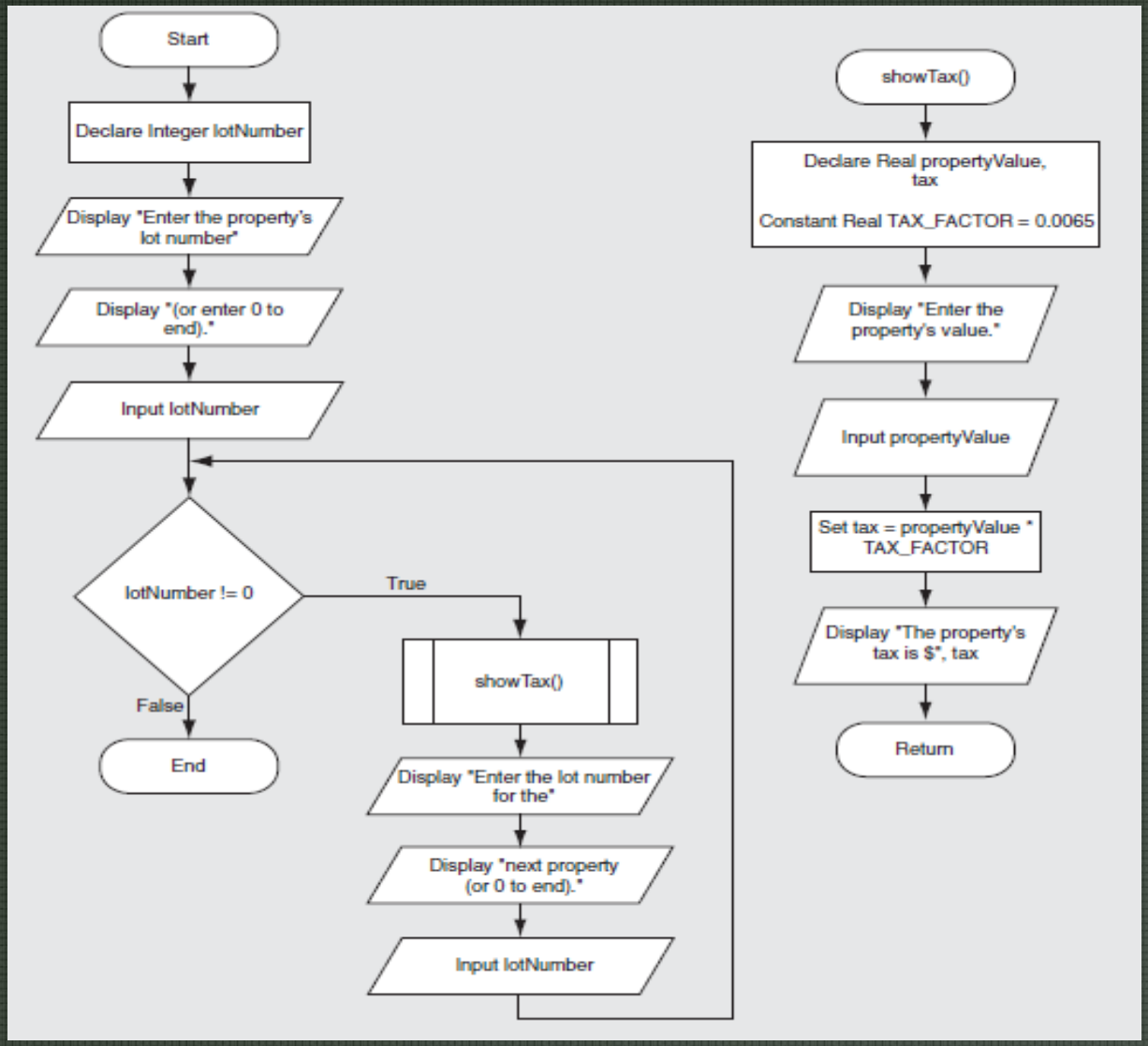
Every day, a clerk in the tax office gets a list of properties and has to calculate the tax for each property on the list. You have been asked to design a program that the clerk can use to perform these calculations.

In your interview with the tax clerk, you learn that each property is assigned a lot number, and all lot numbers are 1 or greater. You decide to write a loop that uses the number 0 as a sentinel value. During each loop iteration, the program will ask the clerk to enter either a property's lot number, or 0 to end.

Chapter 5 – Repeition Structures

```
Module main()
   // Local variable for the lot number
   Declare Integer lotNumber

   // Get the first lot number.
   Display "Enter the property's lot number"
   Display "(or enter 0 to end)."
   Input lotNumber

   // Continue processing as long as the user
   // does not enter lot number 0.
   While lotNumber ! = 0
      // Show the tax for the property.
      Call showTax()

      // Get the next lot number.
      Display "Enter the lot number for the"
      Display "next property (or 0 to end)."
      Input lotNumber
   End While
End Module

// The showTax module gets a property's
// value and displays its tax.
Module showTax()
   // Local variables
   Declare Real propertyValue, tax

   // Constant for the tax factor.
   Constant Real TAX_FACTOR = 0.0065

   // Get the property's value.
   Display "Enter the property's value."
   Input propertyValue

   // Calculate the property's tax.
   Set tax = propertyValue * TAX_FACTOR

   // Display the tax.
   Display "The property's tax is $", tax
End Module
```

Chapter 5 – Repeition Structures



Chapter 5 – Repeition Structures

Nested Loops:

CONCEPT: A loop that is inside another loop is called a nested loop.

A nested loop is a loop that is inside another loop. A clock is a good example of something that works like a nested loop. The second hand, minute hand, and hour hand all spin around the face of the clock. The hour hand, however, only makes 1 revolution for every 12 of the minute hand's revolutions. And it takes 60 revolutions of the second hand for the minute hand to make 1 revolution.

```
Declare Integer hours, minutes, seconds
For hours = 0 To 23
   For minutes = 0 To 59
      For seconds = 0 To 59
         Display hours, ":", minutes, ":", seconds
      End For
   End For
End For
```

If this were a real program, its output would be:

```
0:0:0
0:0:1
0:0:2
```

(The program will count through each second of 24 hours.)

```
23:59:59
```

Chapter 5 – Repeition Structures

Problem:

Design a program for a teacher to get the average of each student's test scores.

```
1  // This program averages test scores. It asks the user for the
2  // number of students and the number of test scores per student.
3  Declare Integer numStudents
4  Declare Integer numTestScores
5  Declare Integer total
6  Declare Integer student
7  Declare Integer testNum
8  Declare Real score
9  Declare Real average
10
11 // Get the number of students.
12 Display "How many students do you have?"
13 Input numStudents
14
15 // Get the number of test scores per student.
16 Display "How many test scores per student?"
17 Input numTestScores
18
19 // Determine each student's average test score.
20 For student = 1 To numStudents
21     // Initialize an accumulator for test scores.
22     Set total = 0
23
24     // Get a student's test scores.
25     Display "Student number ", student
26     Display "-----------------"
27     For testNum = 1 To numTestScores
28         Display "Enter test number ", testNum, ":"
29         Input score
30         Set total = total + score
31     End For
32
33     // Calculate the average test score for this student.
34     Set average = total / numTestScores
35
36     // Display the average.
37     Display "The average for student ", student, " is ", average
38     Display
39 End For
```

Program Output (with Input Shown in Bold)

```
How many students do you have?
3 [Enter]
How many test scores per student?
3 [Enter]
Student number 1
-----------------
Enter test number 1:
100 [Enter]
Enter test number 2:
95 [Enter]
Enter test number 3:
90 [Enter]
The average for student number 1 is 95.0

Student number 2
-----------------
Enter test number 1:
80 [Enter]
Enter test number 2:
81 [Enter]
Enter test number 3:
82 [Enter]
The average for student number 2 is 81.0

Student number 3
-----------------
Enter test number 1:
75 [Enter]
Enter test number 2:
85 [Enter]
Enter test number 3:
80 [Enter]
The average for student number 3 is 80.0
```

Chapter 5 – Repeition Structures

Programming Exercises : (Pseudocode /Flowchart)

1.Calculating the Factorial of a Number

In mathematics, the notation n! represents the factorial of the nonnegative integer n. The factorial of n is the product of all the nonnegative integers from 1 up through n. For example:

7! = 1 × 2 × 3 × 4 × 5 × 6 × 7 = 5,040

And 4! = 1 × 2 × 3 × 4 = 24

Design a program that asks the user to enter a nonnegative integer and then displays the factorial of that number.

2.Celsius to Fahrenheit Table

Design a program that displays a table of the Celsius temperatures 0 through 20 and their Fahrenheit equivalents. The formula for converting a temperature from Celsius to Fahrenheit is

$$F = \frac{9}{5}C + 32$$

where F is the Fahrenheit temperature and C is the Celsius temperature. Your program must use a loop to display the table.