Programming Logic & Design

Chapter 5 – Repetition Structures (Part 1)

Queen's College

CSD 1133 – CPCM -2023S

Chapter 5 – Repition Structures

Topics:

- Introduction to Repetition Structures
- Condition-Controlled Loops: While, Do-While, and Do-Until Nested Decision Structures
- Calculating a Running Total
- Sentinels
- Nested Loops

Chapter 5 – Repeition Structures

Introduction to Repetition Structures

Programmers commonly have to write code that performs the same task over and over. For example, suppose you have been asked to write a program that calculates a 10 percent sales commission for several salespeople. Although it would not be a good design, one approach would be to write the code to calculate one salesperson's commission, and then repeat that code for each salesperson. For example, look at the following pseudocode:

```
// Variables for sales and commission.
Declare Real sales, commission

// Constant for the commission rate.
Constant Real COMMISSION_RATE = 0.10

// Get the amount of sales.
Display "Enter the amount of sales."
Input sales

// Calculate the commission.
Set commission = sales * COMMISSION_RATE

// Display the commission
Display "The commission is $", commission
```

This calculates the first salesperson's commission.

Chapter 5 – Repeition Structures

```
// Get the amount of sales.
Display "Enter the amount of sales."
Input sales

// Calculate the commission.
Set commission = sales * COMMISSION_RATE

// Display the commission
Display "The commission is $", commission
```

This calculates the second salesperson's commission.

And this code goes on and on . . .

There are several disadvantages to this approach, including the following:

- The duplicated code makes the program large.
- Writing a long sequence of statements can be time consuming.
- If part of the duplicated code has to be corrected or changed, then the correction or change has to be done many times.

Instead of writing the same sequence of statements over and over, a better way to repeatedly perform an operation is to write the code for the operation once, and then place that code in a structure that makes the computer repeat it as many times as necessary. This can be done with a repetition structure, which is more commonly known as a loop.

Chapter 5 – Repeition Structures

Condition-Controlled Loops: While, Do-While, and Do-Until

In this chapter, we will look at two broad categories of loops: condition-controlled and count-controlled. A condition-controlled loop uses a true/false condition to control the number of times that it repeats. A count-controlled loop repeats a specific number of times.

Condition-Controlled Loops: While, Do-While, and Do-Until

CONCEPT: Both the While and Do-While loops cause a statement or set of statements to repeat as long as a condition is true. The Do-Until loop causes a statement or set of statements to repeat until a condition is true.

The While Loop

The While loop gets its name from the way it works: While a condition is true, do some task. The loop has two parts: (1) a condition that is tested for a true or false value, and (2) a statement or set of statements that is repeated as long as the condition is true. Figure in the next slide shows the logic of a While loop.

Chapter 5 – Repeition Structures

Condition-Controlled Loops: While, Do-While, and Do-Until

In this chapter, we will look at two broad categories of loops: condition-controlled and count-controlled. A condition-controlled loop uses a true/false condition to control the number of times that it repeats. A count-controlled loop repeats a specific number of times.

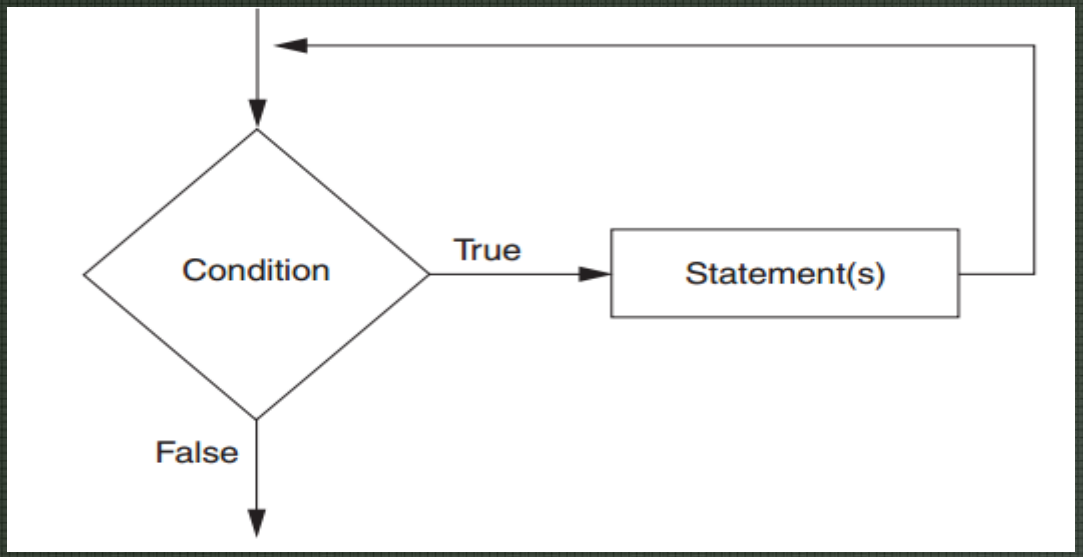
Condition-Controlled Loops: While, Do-While, and Do-Until

CONCEPT: Both the While and Do-While loops cause a statement or set of statements to repeat as long as a condition is true. The Do-Until loop causes a statement or set of statements to repeat until a condition is true.

The While Loop

The While loop gets its name from the way it works: While a condition is true, do some task. The loop has two parts: (1) a condition that is tested for a true or false value, and (2) a statement or set of statements that is repeated as long as the condition is true. Figure in the next slide shows the logic of a While loop.

Chapter 5 – Repeition Structures



The diamond symbol represents the condition that is tested. Notice what happens if the condition is true: one or more statements are executed and the program's execution flows back to the point just above the diamond symbol. The condition is tested again, and if it is true, the process repeats

Writing a While Loop in Pseudocode

In pseudocode, we will use the While statement to write a While loop. Here is the Next slide shows the general format

Chapter 5 – Repeition Structures

```
While condition
     statement
      statement
      etc.
End While
```

These statements are the body of the loop. They are repeated while the condition is true.

In the general format, the condition is a Boolean expression, and the statements that appear on the lines between the While and the End While clauses are called the body of the loop. When the loop executes, the condition is tested. If it is true, the statements that appear in the body of the loop are executed, and then the loop starts over. If the condition is false, the program exits the loop.

As shown in the general format, you should use the following conventions when you write a While statement:

- Make sure the While clause and the End While clause are aligned.
- Indent the statements in the body of the loop

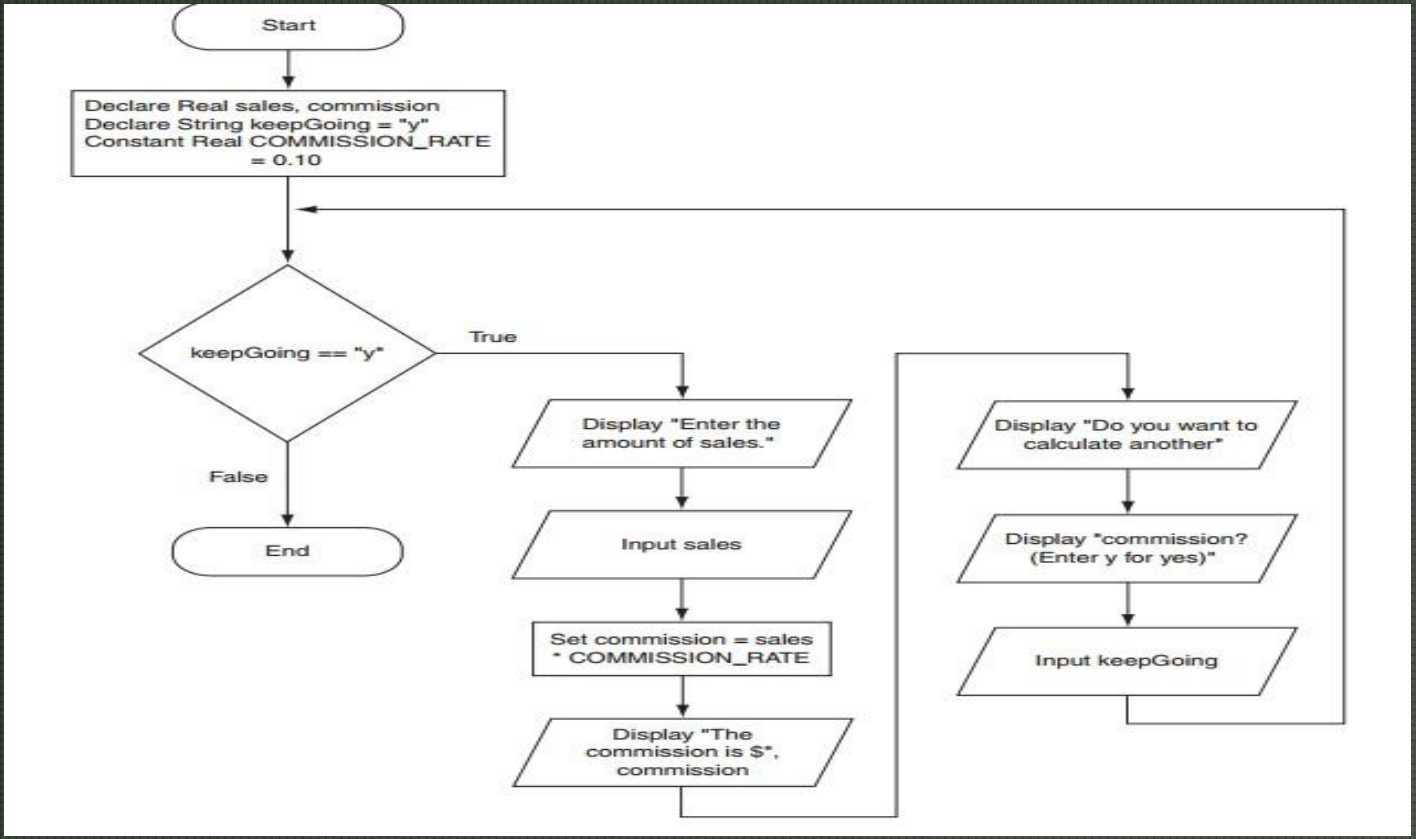
The program show next will show how we might use a While loop to write the commission calculating program that was described at the beginning of this slides

Chapter 5 – Repeition Structures

```
1  // Variable declarations
2  Declare Real sales, commission
3  Declare String keepGoing = "y"
4
5  // Constant for the commission rate
6  Constant Real COMMISSION_RATE = 0.10
7
8  While keepGoing == "y"
9      // Get the amount of sales.
10     Display "Enter the amount of sales."
11     Input sales
12
13     // Calculate the commission.
14     Set commission = sales * COMMISSION_RATE
15
16     // Display the commission
17     Display "The commission is $", commission
18
19     Display "Do you want to calculate another"
20     Display "commission? (Enter y for yes.)"
21     Input keepGoing
22 End While
```

Next we will see the flowchart of the above program.

Chapter 5 – Repeition Structures



The While loop is known as a pretest loop, which means it tests its condition before performing an iteration. Because the test is done at the beginning of the loop, you usually have to perform some steps prior to the loop to make sure that the loop executes at least once.

Chapter 5 – Repeition Structures

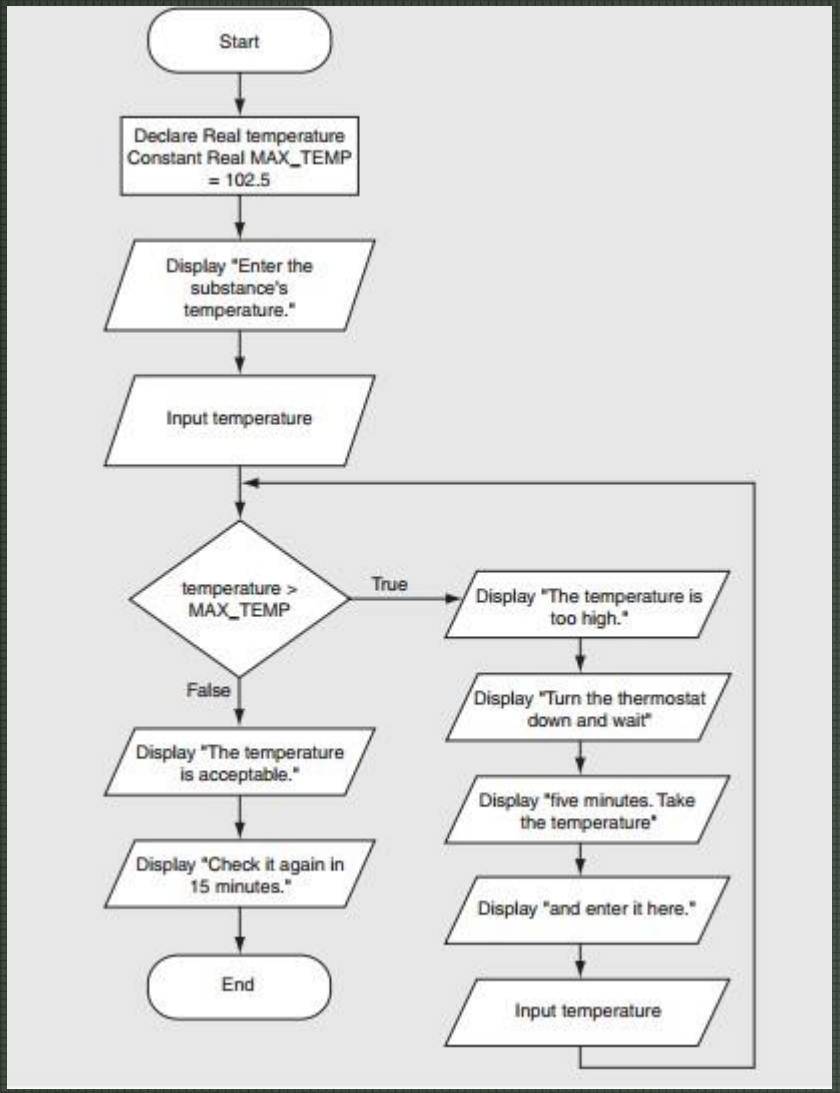
Problem :

A project currently underway at Chemical Labs, Inc. requires that a substance be continually heated in a vat. A technician must check the substance's temperature every 15 minutes. If the substance's temperature does not exceed 102.5, then the technician does nothing. However, if the temperature is greater than 102.5, the technician must turn down the vat's thermostat, wait five minutes, and check the temperature again. The technician repeats these steps until the temperature does not exceed 102.5. The director of engineering has asked you to design a program that guides the technician through this process.

Solution:

```
// Variable to hold the temperature
Declare Real temperature

// Constant for the maximum temperature
Constant Real MAX_TEMP = 102.5

// Get the substance's temperature.
Display "Enter the substance's temperature."
Input temperature

// If necessary, adjust the thermostat.
While temperature > MAX_TEMP
   Display "The temperature is too high."
   Display "Turn the thermostat down and wait"
   Display "five minutes. Take the temperature"
   Display "again and enter it here."
   Input temperature
End While

// Remind the user to check the temperature
// again in 15 minutes.
Display "The temperature is acceptable."
Display "Check it again in 15 minutes."
```

Chapter 5 – Repeition Structures



Chapter 5 – Repeition Structures

In all but rare cases, loops must contain within themselves a way to terminate. This means that something inside the loop must eventually make the test condition false. The loop in shown below stops when the expression keep Going == "y" is false. If a loop does not have a way of stopping, it is called an infinite loop. An infinite loop continues to repeat until the program is interrupted. Infinite loops usually occur when the programmer forgets to write code inside the loop that makes the test condition false. In most circumstances you should avoid writing infinite loops.

infinite loop !

```
// Variable declarations
Declare Real sales, commission
Declare String keepGoing = "y"

// Constant for the commission rate
Constant Real COMMISSION_RATE = 0.10

// Warning! Infinite loop!
While keepGoing == "y"
    // Get the amount of sales.
    Display "Enter the amount of sales."
    Input sales

    // Calculate the commission.
    Set commission = sales * COMMISSION_RATE

    // Display the commission
    Display "The commission is $", commission
End While
```

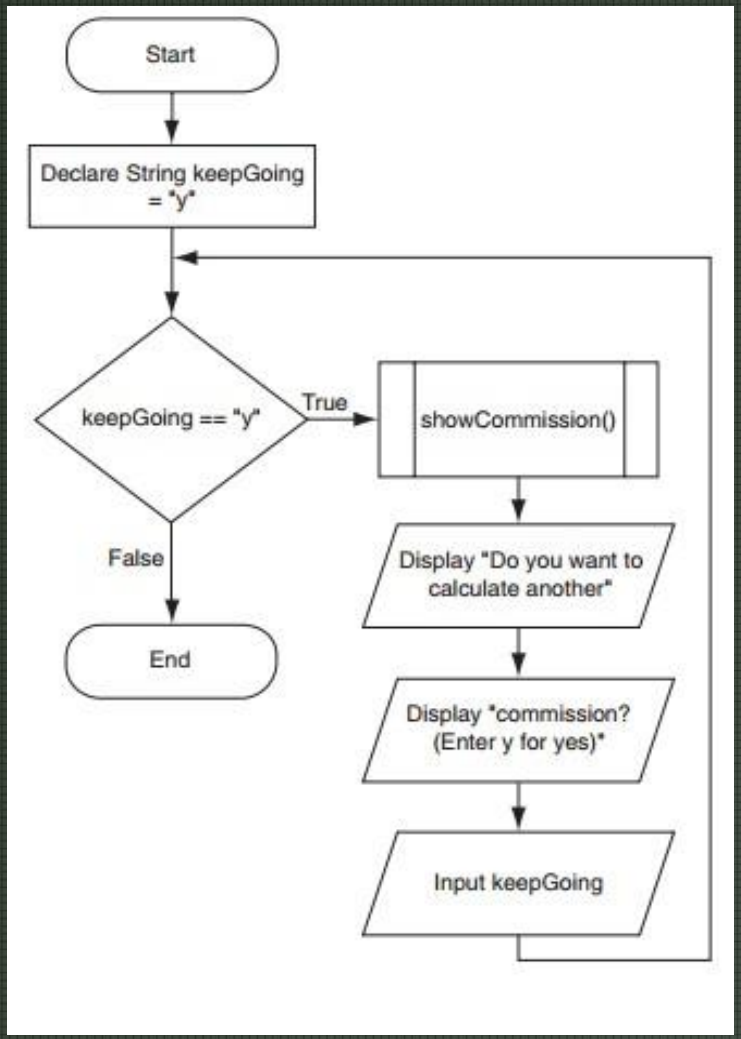
Chapter 5 – Repeition Structures

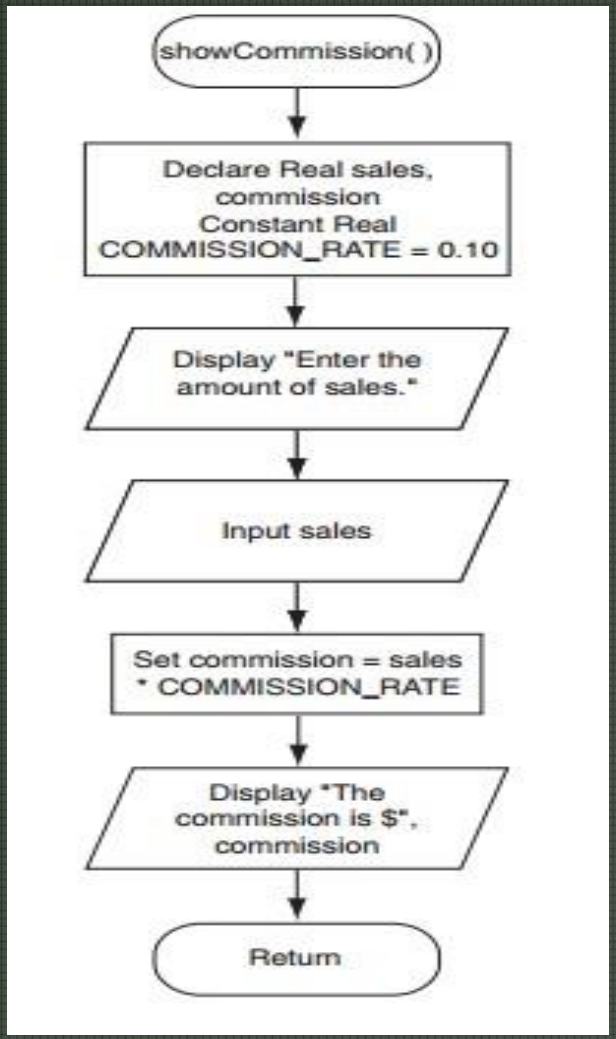
Modularizing the Code in the Body of a Loop

Modules can be called from statements in the body of a loop. In fact, modularizing the code in a loop often improves the design. For example, in Program, the statements that get the amount of sales, calculate the commission, and display the commission can easily be placed in a module.

```
Module main()
    // Local variable
    Declare String keepGoing = "y"

    // Calculate as many commissions
    // as needed.
    While keepGoing == "y"
       // Display a salesperson's commission.
       Call showCommission()

       // Do it again?
       Display "Do you want to calculate another"
       Display "commission? (Enter y for yes.)"
       Input keepGoing
    End While
End Module

// The showCommission module gets the
// amount of sales and displays the
// commission.
Module show Commission()
    // Local variables
    Declare Real sales, commission

    // Constant for the commission rate
    Constant Real COMMISSION_RATE = 0.10

    // Get the amount of sales.
    Display "Enter the amount of sales."
    Input sales

    // Calculate the commission.
    Set commission = sales * COMMISSION_RATE

    // Display the commission
    Display "The commission is  $", commission
End Module
```

Chapter 5 – Repeition Structures

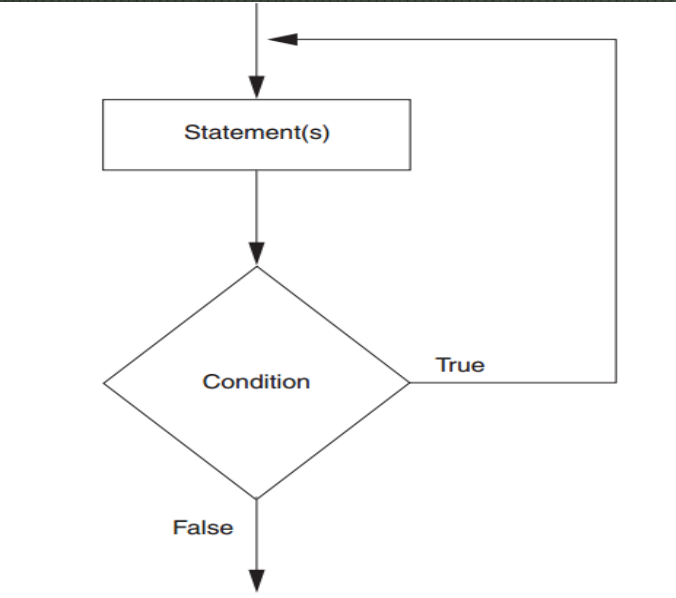




Chapter 5 – Repeition Structures

Do-While Loop:

The Do-While loop is a posttest loop. This means it performs an iteration before testing its condition. As a result, the Do-While loop always performs at least one iteration, even if its condition is false to begin with



Writing a Do-While Loop in Pseudocode

In pseudocode, we will use the Do-While statement to write a Do-While loop. Here is the general format of the Do-While statement:

Chapter 5 – Repeition Structures

Do-While Loop:

The Do-While loop is a posttest loop. This means it performs an iteration before testing its condition. As a result, the Do-While loop always performs at least one iteration, even if its condition is false to begin with

```
Do
   statement
   statement
   etc.
While condition
```

These statements are the body of the loop. They are always performed once, and then repeated while the condition is true.

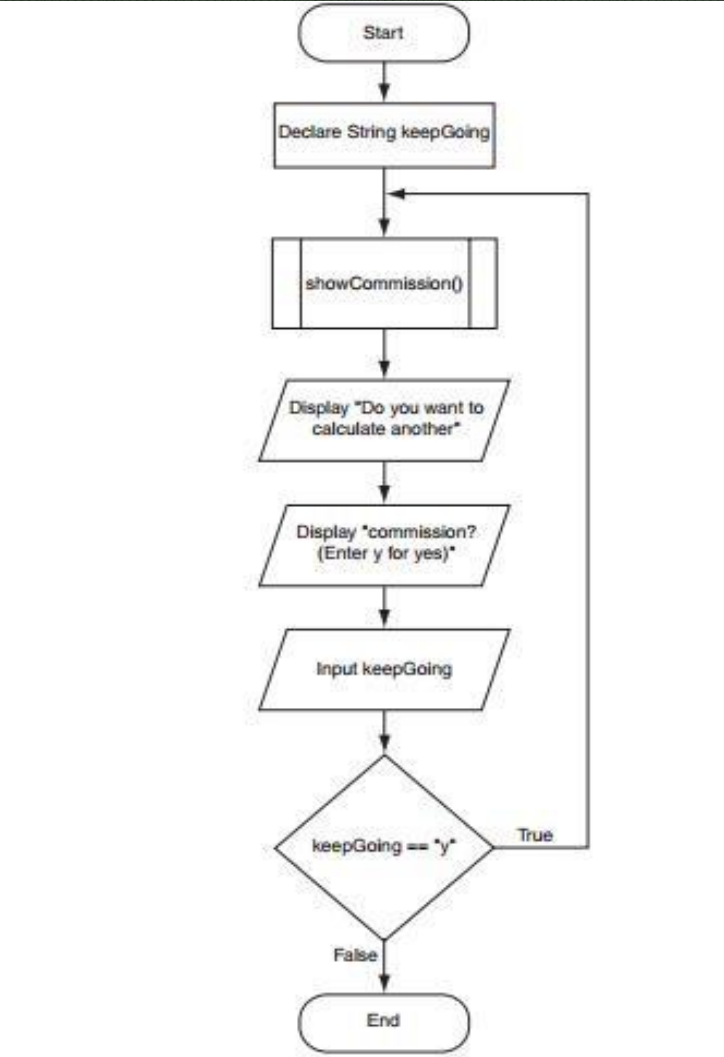
In the general format, the statements that appear in the lines between the Do and the While clauses are the body of the loop. The condition that appears after the While clause is a Boolean expression. When the loop executes, the statements in the body of the loop are executed, and then the condition is tested. If the condition is true, the loop starts over and the statements in the body are executed again. If the condition is false, however, the program exits the loop.

Chapter 5 – Repeition Structures

```
Module main()
   // Local variable
   Declare String keepGoing

   // Calculate commissions as many
   // times as needed.
   Do
      // Display a salesperson's commission.
      Call showCommission()

      // Do it again?
      Display "Do you want to calculate another"
      Display "commission? (Enter y for yes.)"
      Input keepGoing
   While keepGoing == "y"
End Module
```

```
Module showCommission()
   // Local variables
   Declare Real sales, commission

   // Constant for the commission rate
   Constant Real COMMISSION_RATE = 0.10

   // Get the amount of sales.
   Display "Enter the amount of sales."
   Input sales

   // Calculate the commission.
   Set commission = sales * COMMISSION_RATE

   // Display the commission
   Display "The commission is $", commission
End Module
```

Chapter 5 – Repeition Structures



Chapter 5 – Repeition Structures

Problem :

Design a pseudocode and Flowchart for the above

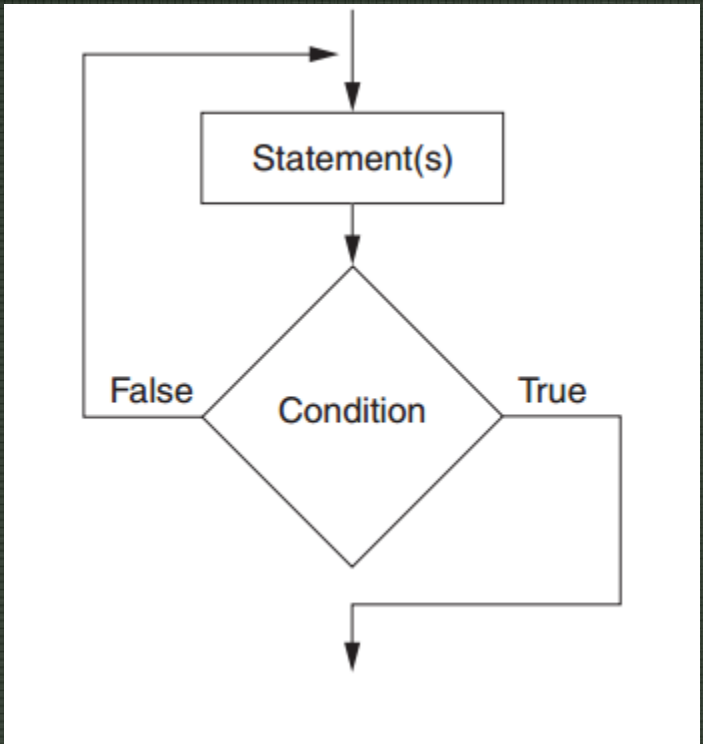
Samantha owns an import business and she calculates the retail prices of her products with the following formula:
Retail Price = Wholesale Cost × 2.5
She has asked you to design a program to do this calculation for each item that she receives in a shipment. You learn that each shipment contains various numbers of items, so you decide to use a loop that calculates the price for one item, and then asks her whether she has another item. The loop will iterate as long as she indicates that she has another item.

For Discussion during the class time

Chapter 5 – Repeition Structures

Do-Until Loop:

Both the While and the Do-While loops iterate as long as a condition is true. Sometimes, however, it is more convenient to write a loop that iterates until a condition is true—that is, a loop that iterates as long as a condition is false, and then stops when the condition becomes true.



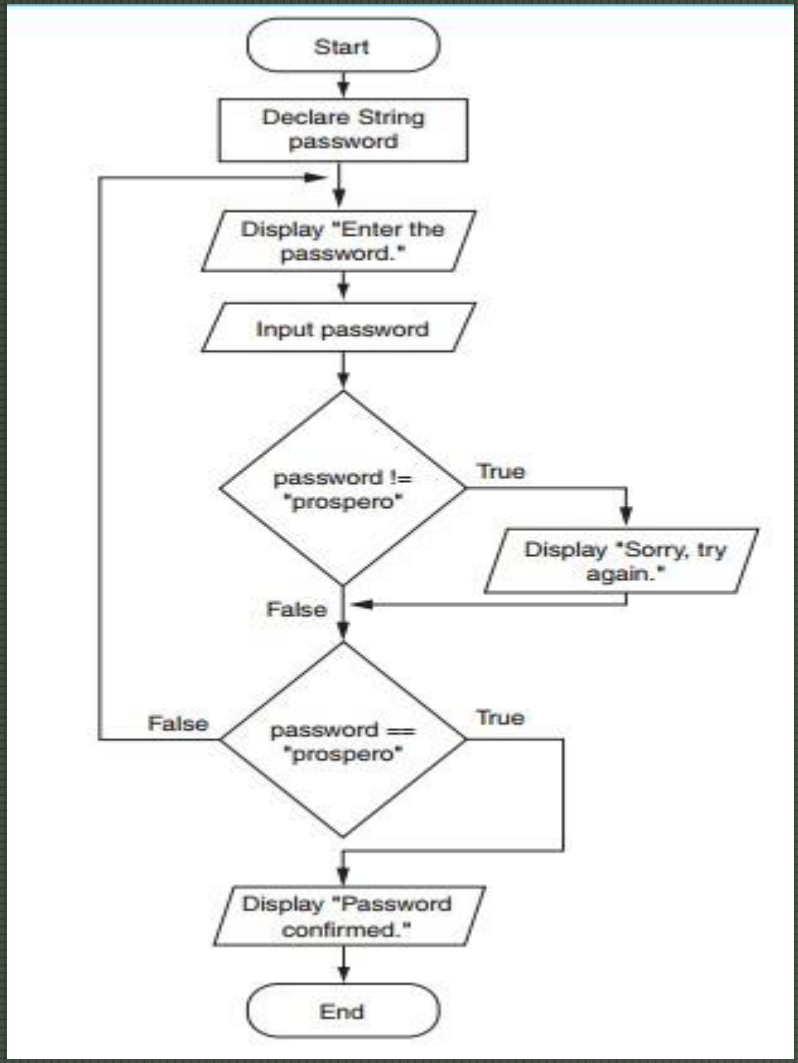

```
Do
   statement
   statement
   etc.
Until condition
```

These statements are the body of the loop. They are always performed once, and then repeated until the condition is true.

Chapter 5 – Repeition Structures

Do-Until Loop:

```
// Declare a variable to hold the password.
Declare String password

// Repeatedly ask the user to enter a password
// until the correct one is entered.
Do
   // Prompt the user to enter the password.
   Display "Enter the password."
   Input password

   // Display an error message if the wrong
   // password was entered.
   If password != "prospero" Then
      Display "Sorry, try again."
   End If
Until password == "prospero"

// Indicate that the password is confirmed.
Display "Password confirmed."
```



Chapter 5 – Repeition Structures

Deciding Which Loop to Use

When you write a program that requires a condition-controlled loop, you will have to decide which loop to use.

You want to use the While loop to repeat a task as long as a condition is true. The While loop is ideal in situations where the condition might be false to start with, and in such cases you do not want the loop to iterate at all.

The Do-While loop is also a candidate in situations where a task must be repeated as long as a condition is true. It is the best choice, however, when you always want the task to be performed at least once, regardless of whether the condition is true or false to start with.

The Do-Until loop also performs a task at least once. It is the best choice, however, when you want to perform a task until a condition is true. The Do-Until loop will repeat as long as its condition is false. When the condition is true, the Do-Until loop stops.

Chapter 5 – Repetition Structures

Checkpoint :

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

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

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

4. What is an infinite loop?

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

Next :

Count-Controlled Loops and the For Statement