CSE 1530

Introduction to Computer Use II: Programming

Winter 2006 (Section M)

Topic D: Control Structures - Iteration

Monday, February 20 2006

Bill Kapralos

CSE 1530, Winter 2006, Bill Kapralos

Overview (1):

- **Before We Begin**
  - Some administrative details
  - Some questions to consider
- **Topic Overview**
  - Introduction/Overview
- **Loops (Iteration)**
  - Introduction/Overview
  - Conditional Loops
  - "Live" Visual Basic examples

Before We Begin

Administrative Details (1):

- **Test 1**
  - Will be distributed after the lecture
  - We will discuss the test in detail at a later time
- **Lab Exercises**
  - You should be working on Ex 5-3 this week
  - Exercise 4-4 will be distributed after the lecture
  - Still have a few previous exercises that were previously distributed & have not been picked up yet
    - If you have not picked up your exercise yet, you can do so after the lecture

---

Some Questions to Consider (1):

- What is a Checkbox ?
- What is an Option control ?
- What is a control array ?
- How do we create a control array ?
- What is the name assigned to a control array ?
- How do we access the elements of a control array ?

Topic Overview

Introduction (1):

- **If Statements Are an Important Alternative to Sequential Processing**
  - However, they are not the only alternative!
  - Another important alternative to sequential processing is iteration (also known as a loop)
    - Iteration → repeatedly executing a set of statements
    - The statements are written once of course and repeated N times

Introduction (2):

- **Until Now**
  - No way to repeat the same statement(s) without calling (e.g., explicitly writing) the statements a second (or multiple) time(s)
  - Computers are however capable of repeating a set of instructions many times → the process of repeating a set of instructions is known as looping
    - An iteration is a single execution of the statement(s) in the loop

Introduction (3):

- **Advantages of the Iteration Construct**
  - Widely used in every programming language and useful in many situations
    - Extremely useful when dealing with arrays → allows us to iterate through the elements of the array using one statement instead of N
  - Allows us to execute one or more statements without having to re-write the statements over and over
    - Of course, there must be some changes made to the values of the variables used in the statements otherwise, the iteration would be pointless!

Introduction (4):

- **Overview of Topic D**
  - We will learn how to write projects (programs) that repeat (iterate) one or more statements
    - How we can go about repeating statements within our code
  - Main concepts of the this topic
    - Setting up iteration → setting up the three components comprising the iteration concept (initialization, termination & loop body)
    - Different types of iteration
    - Manipulating strings using iteration

Loops (Iteration)

Introduction (1):

- **What is a Loop ?**
  - One of the most useful and most used statements in most programming languages, not only Visual Basic!
  - Allows for the repetition of a set of statements
    - Instead of replicating the statements multiple times in the code, it is written once and enclosed within a "loop"
    - Makes code "simpler" and easier to read and follow
    - Can make code shorter

Introduction (2):

- **A Real World Analogy**
  - Consider a stack of bills that need to be paid
    - Need to determine the total amount we must pay

Obtain a stack of bills
Initialize the total to zero
While there is a bill remaining
     Determine the value of the bill
     Add the bill's value to the total
     Remove bill from the stack
Record the total

The three indented statements will be repeatedly executed until all bills in the stack of bills have been added to the total

Introduction (3):

- **Although Simple, the Example Illustrates Three Important Parts of a Loop**
  1. Initialization
  2. Termination
  3. Loop body
- Basically, we start with the initialization, repeat the statements of the loop body and then finally reach the termination
  - After each iteration we come closer to the termination → for example, after each bill's value is added, we have one less bill to consider

Introduction (4):

- **Initialization**
  - Zero or more statements executed once before the first loop iteration
  - May contain variable declarations
- **Termination**
  - Boolean expression evaluated at the start of each iteration
    - If equal to "true" → loop statement(s) executed
    - If equal to "false" → loop statement(s) not executed

Introduction (5):

- **Loop Body**
  - The statements that are executed during each iteration
  - Execution of these statements should bring us closer to the termination of the loop
    - Otherwise, the loop may never end!

Introduction (6):

- **Different Types of Loops**
  - There are several types of loops that can be used
    - One type or another may be better suited for the particular task to be completed
  - In Visual Basic, we will consider two types of loops
    - Conditional loops → loop terminates based on a condition specified by the programmer
    - Counted loops → loop terminates when a pre-specified number of iterations have been executed
  - Lets look at conditional loops in greater detail...

Conditional Loops (1):

- **What is a Conditional Loop ?**
  - Loop statements are executed provided a specified condition is true
    - Allows for the loop to be executed an unknown number of times → we may not necessarily know how many iterations need to be executed
    - Going back to the earlier analogy, we iterate through the loop statements provided there is one or more bills within the "stack of bills"
    - As soon as the last bill has been processed, the next iteration will not be executed since the condition that there are bills in the stack is false

Conditional Loops (2):

- **What is a Conditional Loop ? (cont.)**
  - The idea of executing a set of statements in a program is really not new and you are familiar with it even if you are not really aware of it!
    - Event driven programs are an example of programs that implement a conditional loop → the condition is however controlled by the user

    > Repeat the whole program
    > (until the Exit button is pressed)

Conditional Loops (3):

- **Conditional Loops in Visual Basic**
  - Known as the Do/While Loop
    - Two versions of it
  - General form (syntax) of the Do/While Loops

```
Do [While | Until] (Condition)
    statements of loop body
Loop


Do
    statements of loop body
Loop [While | Until] (Condition)
```
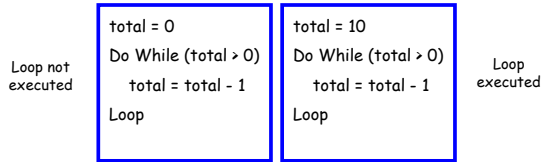
Can use either the "While" or "Until" keyword in either form of the Do/While loop

While → executes loop body provided the condition is true

Until → executes loop body until the point that the condition becomes true
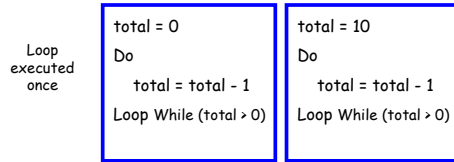
Conditional Loops (4):

- **Conditional Loops in Visual Basic (cont.)**
  - In the first form, the Do/While loop tests for termination at the top of the loop (e.g., before the loop iterates)
    - Statements inside the loop may never be executed if the terminating condition is True the first time it is tested

Loop not executed

```
total = 0
Do While (total > 0)
    total = total - 1
Loop
```

```
total = 10
Do While (total > 0)
    total = total - 1
Loop
```

Loop executed

Conditional Loops (5):

- **Conditional Loops in Visual Basic (cont.)**
  - In the second form, the Do/While loop tests for termination at the end of the loop (e.g., after the loop iterates)
    - Guarantees that at least one iteration of the loop is performed regardless of the condition

Loop executed once

```
total = 0
Do
    total = total - 1
Loop While (total > 0)
```

```
total = 10
Do
    total = total - 1
Loop While (total > 0)
```

Conditional Loops (6):

- **Some Notes Regarding Conditional Loops**
  - Assuming the loop does start, then the loop must do something that will eventually lead to its termination (e.g., condition evaluating to False for the "While" version or True for the "Until" version)
    - This is actually an important aspect of a loop otherwise, you could have a situation where a loop executes "forever" → infinite loop
    - It is up to you as the programmer to ensure you do not set up an infinite loop!

Conditional Loops (7):

- **"Live" Examples of Loops**
  - Lets look at some simple examples of working with conditional loops in Visual Basic