Chapter 7a
Iteration (looping)

Chapter Objectives

- Understand the necessity of this program control structure
- Describe while loops
- Describe do…while loops
- Describe for loops

Why Use A Loop?

- Repeat instructions with many data sets
  - Repetition or iteration structures
- Rich set of looping structures
  - while
  - do…while
  - for
  - foreach statements
Using the while Statement

- Simplest and most frequently used loop
  while (conditional expression)
  statement(s);
- Expression – sometimes called loop condition
  - Returns a Boolean result of true or false
  - No semicolon after the conditional expression
  - Null body – empty bodied loop – infinite loop
- Enclose multiple statements for body in {    }

while Statement

- Pretest
  - If the conditional expression evaluates to true, statement(s) performed
  - If the conditional expression evaluates to false, statement(s) skipped

Counter-Controlled Loop

- Loop control variable
  - Variable simulating a counter
    - Initialized
  - Conditional expression designed so that you can exit the loop after a certain number of iterations
  - Increment counter with each iteration
    - Otherwise, infinite loop
Counter-Controlled Loop Example

```csharp
int sum = 0;
int number = 1;
while (number < 11)
{
    sum = sum + number;
    number++;
}
messageBox.show("Sum of values "
    + "1 through 10"
    + " is " + sum, "Example");
```

Counter-Controlled Loop (continued)

- Common problem
  - Off-by-one error
    - Loop body not executed for the last value OR
    - Loop body executed one too many times

Sentinel-Controlled Loop

- Exact number of times loop body should execute not known
- Often used for inputting data
  - Prime read on outside of loop
- Also referred to as indefinite loops
- Select a sentinel value
  - Extreme value or dummy value
  - Sentinel value used as operand in conditional expression
  - Tells user what value to type to end loop
Sentinel-Controlled Loop (continued)

- Useful for loops that process data stored in a file
  - Sentinel is placed as last entry in file
  - Conditional expression must match selected sentinel value
- See class demo for an example

State-Controlled Loops

- Similar to sentinel-controlled loop
  - Referred to as flag-controlled loops
- Instead of requiring a dummy or extreme value, use flag variable
- Can be Boolean variable (not a requirement)
  - Variable must be initialized
  - For each new iteration, evaluate to see when it changes state
  - Change its value inside the loop – to stop the loop

State-Controlled Loops Example

```csharp
bool moreData = true;
while (moreData)
{
    // moreData is updated inside the loop condition changes
    if (MessageBox.Show("Do you want another number?", "State Controlled Loop", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No)
    {
        // Test to see if No clicked
        moreData = false;
    }
    // End of if statement
    // More loop body statements
}
// End of while loop
```
do...while Loop

- Very similar to the while loop
- A “do” block of code is executed first
- After the block executes at least once, then a while statement is tested
- Example:
  ```csharp
double sum = 0;
int counter = 0;
do {
    sum += sum
    counter ++
} while counter <= 10; //note: this text falls outside the loop itself
```

for Loop

- Pretest form of loop (like the while)
- Considered specialized form of while statement
- Usually associated with counter-controlled types
- Packages initialization, test, and update all on one line
- General form is:
  ```csharp
  for (statement; conditional expression; statement)
  {
    statement(s);
  }
  ```

For Loop (continued)
CIS 3260 – Intro to Programming in C#

Chapter 7a – Page 6

McGraw-Hill © 2010 The McGraw-Hill Companies, Inc. All rights reserved.

int counter = 0;
while (counter < 11)
{
    MessageBox.Show(counter + "", " +
Math.Pow(counter,2) + ", " + Math.Pow(counter,3));
    counter++;
}

for (int counter = 0; counter < 11; counter++)
{
    MessageBox.Show(counter + "", " +
Math.Pow(counter,2) + ", " + Math.Pow(counter,3));
}

Replace above while loop with for loop below -- does same

Endless Loops

- Never change the value of the index variable inside the loop
  - Poor programming practice
  - May lead to an endless loop

// Poor Programming.
For (int indexInteger = 1; indexInteger < 10; indexInteger++)
{
    indexInteger = 1;
}