Week of January 25, 2015

Topics for this week: Expressions

Activity Checklist

	Read chapter 4 in your course packet.
	Review the slides on Expressions
	Review the sample program Expressions
	Review the video Expressions
	Review the video Casts
	Complete lab #6, due by 11:59pm on Tuesday.
	Complete lab #7, due by 11:59pm on Thursday.
	Complete project #2 and submit it to Canvas before 11:59pm on Sunday. Programs will lose 20% of the possible points for each day that they are late. If you turn this program in prior to 11:59pm on Saturday, you will receive a 5 point bonus, **if** it meets all of the specifications and gives the correct answers.
	Start to study for the first exam. It is coming up next week.

Learning Goals

It is expected that you will meet the objectives outlined here by the end of the week. You might want to test yourself to see how well you fare. You can be guaranteed that you will be tested on these concepts on your first midterm. By the end of this unit, you should be able to:

- Correctly use the basic arithmetic operators in a C# program.
- Describe and correctly use C#'s precedence rules.
- Describe integer and floating point division and correctly write expressions using each.
- Show how automatic data conversions are performed in mixed expressions.
- Tell what a type cast is and correctly use a static_cast in a C# program.
- Correctly use the increment and decrement operators in a C# program.
- Explain the difference between post- and pre-increment or decrement.
- Correctly use the String class in a C# program.
- Correctly use the String processing functions of the string class.
- Follow the steps for problem solving outlined in lab #2 to solve computing problems.
- Create UML activity diagrams to show the steps involved in a solving a problem.

Reading Assignment

All reading should be done before you come to class. Your ability to understand the material discussed in class will be greatly enhanced when you come to class prepared.

1. Chapter 4 in your course packet explains how to write expressions in C#. It also introduces some of the built-in math functions, which we will talk about in a few weeks. This chapter provides some helpful programming hints and points out some common errors students make when first writing expressions.

2. The slides on Expressions introduces the arithmetic operators and explains how to write arithmetic expressions. One of the most difficult concepts presented in the slides is the handling of mixed data types. Be sure that you understand this important topic.


Don't pass over this material lighty. There are some subtle concepts here that must be mastered. You will be tested over and over again on problems that involve mixed data types, precedence and integer division.

Key Concepts

Be sure that you understand the following important ideas presented in this unit.

1. C# provides all of the common arithmetic operations, but in some cases the symbols used are different from what you might be used to. In particular, the multiplication symbol is an asterisk (*) and the division symbol is a slash (/). In addition, C# provides a number of "shortcut" symbols to do often used operations. For example, incrementing a value is done so often in C# that the developers of the language created a new symbol, ++, to signify the increment operation.
2. When an expression in C# is evaluated, evaluation follows a fixed set of **precedence rules**. These rules define which operations get done first. For example, in the expression

   ```
   num = factor * size - width;
   ```

   the multiplication operation will be done before the subtraction. If you are not sure about the precedence rules, or if you want to change the order of operation, you can use parentheses, just like you would in algebra.
3. One of the most important concepts in this module is the handling of mixed data types in an expression. For example, what happens in the expression

   ```
   area = length * width;
   ```

   if *length* is an integer and *width* is a double? The computer hardware does not know how to multiply an integer by a double. In order to do this processing, the computer must make both *width* and *length* the same kind of data. There are precise rules that are used to make these conversions. It is important that you understand these rules.
4. Occasionally, a programmer will need to make an explicit change of data type. This is done with a **cast**.

Lab Assignment

This week you should complete labs 6 and 7. These labs will help you to understand how expressions are evaluated, and go though the important topic of developing an algorithm. Be sure that you spend sufficient time studying this these ideas, and practice them during the week. Be sure that you know how to write pseudo-code.

Programming Project

This week you should complete your second programming project. In this project, you will create a program that uses expressions. This will give you practice using the concepts that have been discussed this week.

Be sure that you include the declaration that you did not copy any code in all of your source code files. If this statement does not appear in your program, it will not be graded. All program are expected to contain the pseudo-code that you created when designing the solution to the problem.