Programming Example 4: Expressions

Introduction

This example illustrates the use of an activity diagram to show the logic flow in a program, and points out a number of important principles student's should keep in mind when writing simple arithmetic expressions.

As you read through this example, take note of the following important principles:

1. **Integer Division:** When doing division, if both the numerator and the denominator are integers, then the quotient will be an integer. This occurs because the division is done in integer hardware in the computer. There is no place to store any fractional data.
2. **Pre and post increment:** There is both a pre-increment and a post-increment operation. Pre-increments are done before anything else in the expression, post-increments are done after everything else in the expression.
3. **Remainder Operator:** One of the interesting operators introduced this week is the remainder operator. The expression *a = b % c* divides *b* by *c* and returns the remainder. This operator is surprisingly useful.
4. **Precedence:** In an expression containing several arithmetic operators, multiplication and division always takes place before addition and subtraction.
5. **Mixed mode arithmetic:** When computers do arithmetic, they like to have all of the operands be of the same type. Thus, if two operands are of a different type, for example an integer and a float, the computer will convert one of the operands to match the data type of the other before doing the arithmetic. When a conversion is automatically done, it will always be a widening conversion. The computer will never do a narrowing conversion automatically.
6. **Type casting:** When you want to force the computer to do a conversion, you can do a **cast** operation.

The problem statement for this program is located here.

The activity diagram for this program is located here.

The example program is located here. An executable of this program can be found here.