Programming Example 6: Loops

Introduction

This example shows an activity diagram that contains a loop, and demonstrates a number of important ideas student's should keep in mind when writing programs that contain loops.

As you read through this example, look for code that illustrates these important principles:

1. **Loops:** When you see words such as *count* or *repeat* in a problem description, it is likely that the problem will require one or more loops. There are three basic looping constructs:
2. **while(condition) { ... }:** While loops will repeat as long as the given condition is true. If the condition is initially false, the body of the while loop will never execute.
3. **do { ... } while(condition):** Do-while loops will repeat as long as the condition is true. The body of a do-while loop is guaranteed to execute at least one time. Do-while loops are useful for testing a sentinel, that is, the condition of some variable entered by the user inside of the loop. Note how this construct is used to validate the user's input as either 'y' or 'n'.
4. **for (initialization; condition; increment) { ... }:** For loops are used most often when a loop must repeat a specific number of times. They will repeat as long as the condition is true. Such loops are often called counting loops, because they loop a specific number of times. See how this construct is used in the loop that tests characters in the string to see if they are a one or a zero.

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.