Assignment 1: Single-Linked Lists

Objectives:

- Students will implement a binary number class using a single-linked list.

Background:

For this first assignment, you will use a single-linked list to represent a binary number. The decimal number 37 can be represented by the binary number 100101. This binary number can be shown as 2^5 + 2^2 + 2^0 = 37 where we are adding terms for those bits of the binary number that were 1, and each term is 2 raised to some power dependent on the bits position. Each term can be represented by a node in the linked list storing that bit's position. So the above binary number could be represented by a linked list of (0)->(2)->(5). We are NOT storing 0 bit terms.

You need to use a linked list for this, and you must implement the list 'by hand'; you cannot use the STL list class or similar classes.

You are provided a header file that gives the linked list node structure and a set of methods you must implement. You are also provided a tester (application) file to test the implementation of your class. While the first three steps of the directions are for setting up a project in Visual Studio, you can use which ever IDE and compiler you desire to work on the assignment.

You may work on the assignment with a partner. If you do, make sure you include their name in the comment blocks at the top of each submitted file.

Directions:

1. Open Visual Studio 2010 and select C++ Development Settings. Create a new **empty** C++ project for the lab called **Assignment1**.
2. Download the files binary.h and binary_main.cpp from D2L into your project folder (**Documents/Visual Studio 2010/Projects/Assignment1/Assignment1**).
3. In the solution explorer in Visual Studio, right click the project (**Assignment1**), click add->Existing Item. Select the 2 copied lab files and add them.

LINKED LIST

4. Create the binary.cpp file and add entries for all methods or functions from the header file that are not already implemented.

   NOTE: If you pay attention, you will see the default constructor is already implemented.
5. Implement the logic for all functions/methods in the binary.cpp file. Give some thought to the efficiency of your code. For example, consider if one of the methods loops over the list multiple times and if instead you can combine the operations in one loop.

   HINT: Much of the code is simple if you store your nodes in sorted increasing order.
6. Compile and Run the test code and determine if there are any errors in the code. Correct any errors until the code compiles successfully and runs with expected output.

Turn In (Due date 11:59pm Friday Feb. 22):

- Submit source code for the binary number class (binary.h and binary.cpp).
- Include an info comment block at the top of your files as described in syllabus.
- Comment your code within your methods explaining your logic.