CIS 230 – Spring 2006
Homework #11
Due Friday, May 5, 2006 at 5:00 PM

Submit your files using the submission tool on the qs-server:
/class-files/gilden/230submit

Files to Submit:

- set.h
- set.cpp
- set_test.cpp (unmodified)

Assignment:
Implement a **`Set`** class. Represent a Set using a linked list of a structure, **Item** which has two variables: a number, and a pointer to another Item. (In your Set class, include a pointer to an Item as its private data member, and other variables/pointers as appropriate). For this assignment, any positive number may be used in the Set. Your linked list will hold the elements of the Set, and will be allocated dynamically. It will only contain sufficient cells to represent the integers actually in the set. Thus for {3, 5, 27} the set would be represented with a linked list of 3 Items. In the first Item would be the value 3; in the second Item would be the value 5; in the last Item would be the value 27. Order DOES matter: You will need to use an ORDERED linked list. An empty Set will be represented by a pointer whose value is NULL.

The methods needed for Set class are as follows:

- **void create_set();**
  This method will be used to initially populate the two sets.
- **void addElement();**
  This method is for adding element(s) to the set. You must determine whether the value to be added is already present. If it is, you have nothing more to do. If it isn't, you'll need to dynamically allocate memory for the new Item.
- **void deleteElement();**
  This method is for removing element(s) from the set. Similar to addElement();
- **friend int operator==(Set &, Set &);**
  This **friend** function returns a 1 if the two sets are equal and a 0 if the sets are not equal.
- **friend int subset(Set &, Set &);**
  This **friend** function returns a 1 if the first set is a subset of the second and a 0 if not. If we have two sets {3, 4, 7,10} and {2, 3, 4, 7, 9, 10}, the first set is a subset of the second set.
- **friend Set setUnion(Set &, Set &);**
  This **friend** function returns a Set that is the union of the two parameter sets
- **friend Set intersection(Set &, Set &);**
  This **friend** function retunrs a new set that is the intersection of the two parameter sets.
- **friend Set difference(Set &, Set &);**
  This **friend** function returns a set that is the difference between the first set and the second set. If we have two sets {3, 4, 7,10} and {2, 3, 4, 7, 9, 10}, the difference would be {2, 9, 10}.

- **void report();**
  This method prints out the elements in the set in standard set notation.

You will also need a constructor function. Your constructor function should initialize any pointers that you deem appropriate.

You may add other methods as you deem appropriate.

Your program must interface with the program on the next page (and located on the qs-server at: /class-files/gilden/hw10/set_test.cpp). All inserting, adding, and deleting of the member data must be terminated by a sentinel (-1). Be sure to state in your program what terminates the loop.

Be sure to have meaningful variable names and instructive user prompts and output labels.

Use the following filenames:
- set.h – should contain your Set class declaration
- set.cpp – should contain your Set class definitions
- set_test.cpp – do NOT modify this file.