CS 100
Lab 7: STL and templates

Reminder: find a *new* partner for this lab (50% grade reduction for working with a person ever worked before).
You are allowed to use 1 (one) computer per team (50% grade reduction for using more than 1 computer per team including laptops and tablets).

Grading: 18% each program, 10% attendance.

Do <u>5 out of 6</u> exercises to get full credit and create a MAKEFILE that compiles all 5 of them.

The STL library extends the capabilities of standard C++ with many libraries that bring together data structures and algorithms often used in computer science. So far, at a minimum, you have used the vector class which allows for a simple way to implement dynamic arrays. This lab will contain exercises using other features of the STL library. Although there are too many to go over in a short period of time, practice with a few of them will make it easier to understand the library as a whole as many concepts are repeated throughout the different libraries.

Complete the following exercises for credit:

1. Create a vector container containing some number of values; i.e., create an instance of the STL vector class (you do not need to actually implement your own version of vector). Instead of using the [ ] operator, use iterators to access elements of the vector (take note of various types of iterators available). Use each of the operations insert, erase, swap, clear, at least once to modify the vector.

2. Use a vector container as the basis for the STL stack container adaptor. Use the available operations in the stack container adaptor to take in any number of characters and output them in reverse.
    NOTE: it may be helpful to Google what a "STL container adapter" is.

3. Use a deque (unexpectedly pronounced as "deck") container as the basis for a priority queue container adaptor. Use the priority queue to create a simple scheduler to be used for a list of processes which, given a job id and its priority for each process, orders them such that the job with the greatest priority is at the top of the queue.
    HINT: you may want to look at how to create your own comparator (aka. "Comparison class").

4. Create a wrapper class called my_set, with a vector container as its data member, which gives it the functionality of a mathematical set (inclusion, union, intersection, difference). Use the algorithms provided by the STL library to accomplish this; e.g., set_union, set_intersection, etc.
    NOTE: each function should accept a my_set parameter.
    NOTE: if you are confused as to what inclusion means this is the first Google hit for "set inclusion": http://en.wikipedia.org/wiki/Subset. This function should accept another set and

return a boolean value.

5. Use a map container to create a phone book class. The key for each entry ought to be the full name of the person and the value will be their phone number. The class should be able to insert new entries, delete entries, find a specific entry, and print all entries.

6. Use a graph data structure to model a simple computer network. Each node should be a computer in the network, each computer having a unique id, and there is an edge between two nodes if there is a physical connection in the network between the two corresponding computers. Use a map container to store the graph as an adjacency list. Implement a search method which, when given a starting node and ending node, will determine if there exists a path between them in the network.

Submission:

- Create a test harness for each of these exercises in order to demonstrate that they work, and demonstrate this to the TA.
- Put them in all in one archive and submit it on iLearn.

Useful Resources:

- Guide to STL library:
  - http://www.sgi.com/tech/stl/
- Useful guide to member functions of STL and examples:
  - http://www.cplusplus.com/reference/stl/