Assignment #1
CS 163 Data Structures

Submit your assignment to the D2L Dropbox (sign on via d2l.pdx.edu)

****Assignments in CS163 consists of written homework and programming****
****All parts are required to get a grade on this homework ****

HOMEWORK QUESTIONS:

1. **Create an Algorithm** for the process of how to be successful with this course. Think about what you will need to do this term, in a step by step fashion. Think of (a) what you will need to do (read the syllabus, attend or watch lectures, perform the pre-labs, read the book, practice linear linked lists) (b) be specific about when you need to do each of these (plan ahead), and then (c) think about how best you learn and what you will need to do to take what is discussed or taught and be able to apply it (practice concepts from lectures and the book, transcribe your notes, re-watch lectures?). If you took CS162, what worked for you the best or what could you have done things better? Think about how confident you feel about classes, pointers, linear linked lists and recursion and what you will need to do for review of these topics. Once you have thought about this, write it down as an algorithm in a step by step fashion.

   Algorithms should be written using complete sentences in a way that is easy to follow – consider using outline form for clarity.

2. **Think about Style** *Background: With programming, we can add components to a program to enhance its readability. This may include blank lines, indentation, comments, etc.* Take a look at a C++ program that you have written previously (such as from CS162). How readable and/or understandable is it? Are there things you would do differently (in terms of style) now that you have stepped back from it? **Then, write about three things** that could have been done to make it more readable and maintainable and show the "before" and "after" code *(this means to show the code you think could be improved – and then show the code reworked, improved for the purposes of style).* Use complete sentences.

3. **Ethics.** Using examples from your own experience, discuss how issues of security and identity theft when using smart phones and Wi-Fi connections around town. Research if there are things we need to take into account when using public Wi-Fi.

Programming Project

Scope: When beginning with this project, the first thing to keep in mind is that we only have approximately 2 weeks to complete each assignment. Therefore, it is critical that you focus on a limited scope. *You will be primarily graded on your use of classes, member functions, arguments, data structures, pointers and the efficiency of your code.* Therefore, focus on how to design classes that are well structured and efficient and on the required data structures. Limit the development of the application that uses the data structures. Of course, your user interface must be clear enough for us to test your program and we must be able to thoroughly test all features.

This first program of the term is an exercise in building, traversing, and destroying linear linked lists. In the real world, you would use external data files – but given how little time we have for each program think of that as a future step to the program (done later).

Programming – Problem Statement: Have you ever thought much about what goes on when we compile programs? Programming languages are a particular area of interest for me. I find them fascinating! Consider the following problem faced in the development of a compiler. The source program contains many character string symbols (if, while, for, etc.) as well as variable and function names. Each of these is associated with a memory location, data type, or operation. However, it would be too time consuming to actually do character manipulation each time when working with one of these. Instead, each string may be identified with an integer and viewed as being equivalent to the string for the purpose of compiler manipulation.

Your job is to implement a mapping that will take a name and map it to an integer which in turn will serve as a pointer into a table of attributes for that item, called a symbol table. Each item in the table will contain (a) name, (b) type of item (variable, function, operation, etc.), and (c) the memory location association. We will also add a description of what it is being used for as well (like a comment), just for fun!

Now let's talk about data structures. The best data structure would be an array, because of direct access. But, since we are including all identifiers (including variable and function names), the size cannot be determine ahead of time. Therefore, our data structure for program #1 is a linear linked list of arrays. Each node will contain an array of items. Once the first array is used (all indices used for symbol table entries), add a new node with another array, and continue until the user is done adding entries.

Once the list is built, allow the user to (a) add more, (b) display all symbol table entries, (c) lookup an entry by number, and (d) remove all entries (destructor).

Things you should know...as part of your program:

1) Do not use statically allocated arrays in your classes or structures. All memory must be dynamically allocated and kept to a minimum!

2) All data members in a class must be private

3) Never perform input operations from your class in CS163

4) Global variables are not allowed in CS163

5) Do not use the String class! (use arrays of characters instead!) You may use the cstring class.

6) Use modular design, separating the .h files from the .cpp files. Remember, .h files should contain the class header and any necessary prototypes. The .cpp files should contain function definitions. You must have at least 1 .h file and 2 .cpp files. **Never "#include" .cpp files!**

7) Use the iostream library for all I/O; do not use stdio.h.

8) Make sure to define a constructor and destructor for your class. Your destructor must deallocate all dynamically allocated memory.

9) Remember that 20% of each program's grade is based on a written discussion of the design. *Take a look at the style sheet which gives instruction on the topics that your write-up needs to cover.*