Programming Assignment #4
CS 163 Data Structures

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

Programming – Goal: The goal of this program is to create a binary search trees (BST) and to implement the insert, retrieve, retrieve a range worth of information, display in sorted order, remove all matches by keyword, and remove all. All of these functions should be implemented **recursively.**

Background: The advantage of a binary search tree is the ability to retrieve our data using a logarithmic performance assuming that the tree is relatively balanced and be able to search for a range of information and obtain our data in sorted order. In program #4 we will experience all of these characteristics. Instead of using a hash table, we will use the same concept as Program #3 but this time with a binary search tree, for our Table ADT (by the value of the data). **PLEASE NOTE – there is NO HASH TABLE in program #4.**

Specifics: In Program #4 we will use a BST to search for the keyword that we are interested in. Now we also want get the keywords listed in sorted order and display a range of keywords (from those that start with K through M for example). We will no longer be searching by chapter in this phase of the project. Each item in our tree will have the following information (at a minimum) which should be stored as a struct or a class:

1) Keyword (e.g., hash function)
2) Short description of what you have learned about that topic
3) Chapter number(s)
4) Page number(s) (allow for a list of page numbers that you think are most important)
5) Flag to indicate if additional study on this material is needed

The **Table ADT operations that must be performed on this data are:**
1) Insert a new keyword with its information
2) Remove a keyword with its information
3) Edit the description to add more text as you study, given the keyword
4) Retrieve the information about a particular keyword (only 1 match expected) (Remember, retrieve is NOT a display function and should supply the matching information back to the calling routine through the argument list)
5) Display all keywords within a given range (staring first letter of the keyword to ending first letter of the keyword (e.g. 'K' through 'M').
6) Display all keywords (in sorted order this time)

Data Structures: Write a C++ program that implements and uses a **table ADT (by value) using a binary search tree**. The binary search tree should be a non-linear implementation (using left and right pointers). The underlying data may be stored as a struct or class.

Evaluate the performance of storing and retrieving items from this tree. Monitor the height of the tree and determine how that relates to the number of items. If the number of items is 100 and the height is 90, we know that we do not have a relatively balanced tree!! Use the information from the Carrano reading to assist in determine if we have a reasonable tree, or not. **Your efficiency write up must discuss what you have discovered.**

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 and the cstring library!)**
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.