CSC221 – Laboratory #2*

David John

Fall 2008

This lab will create an interactive program that will read strings from a file into a binary tree. The string **must** be normalized by placing the letters into lower case. The program will process a number of interactive commands to perform different functions. The *string* and *cstring* classes will both be used in this program.

The objects to be inserted into the binary search trees consist of two members, a string and a frequency. You must construct a simple class to contain and manage this information.

For a maximum grade of 75, your program must be able to handle the commands of ***add*** *filename*, ***stop***, and ***print***. The command ***add*** *filename* will cause the contents of the file, *filename*, to be added to the binary search tree. The commands ***print***, will print to the screen the contents of the tree in lexicographical order along with the frequency associated with the strings. The command ***stop*** will cause your program to exit.

For a maximum grade of 85 you must add to the 75 point assignment a command ***bottomup***. This command will traverse the contents of the tree by levels, starting at the bottom and moving up to the root. You will want to use your stack and queue classes. We will discuss this further in class.

For a maximum grade of 100 you must use an *AVL* binary search tree. The **print** command must also print the height of the subtree rooted at the string.

The input to your program will be text. You will be placing into your tree the words of your text with the number of occurrences of each. The

*Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and I am aware of a trademark claim, the designations have been printed in initial caps, all caps or italics.

first time the word is placed in the table, its number of occurrences is 1. All subsequent accesses for that word will update its number of occurrences. Please note that the Internet is a rich source for text!

There **must** be a functional *makefile* to properly compile your program. Your program must be split among appropriate file**s**. I will supply you with a *makefile* that you can use and/or edit.

I will also supply a binary search tree template that you can use.

This assignment will require some time to complete. I have absolutely no pity when you decide to wait to the last minute to start your work. If you decide to wait to the end to start, you are entirely on your own! If you work on this project for one hour each day, concentrating on one class at a time, you will complete this in plenty of time.

Your program is due no later midnight, Friday, October 17, 2008.

This is a pledged work assignment. You must work on this assignment by yourself. You may not collaborate with anyone else[1]. If you have questions about your work, you must ask me[2]. Beware, I have little sympathy for late work. Being a student is your full-time job, it is important that you work full-time as a student.

[1]You may consult with anyone you wish concerning *C++* syntax questions.

[2]I will not answer questions concerned with *C++* syntax unless that are unusually interesting. I am more than willing to talk with you about conceptual issues, *make*, and linking and loading problems.