CS 305 Lab 8: Trees
Fall 2019

The purpose of this lab is to give you experience with binary search trees. In this lab, trees are implemented as recursive data structures.

This lab has a total of 100 possible points (30 points from pre-lab and 70 points from lab).

Sit with your assigned partner.

Objectives

Upon completion of the laboratory exercise, you will be able to do the following:

- define a function to determine the height of a tree
- define a function to do a postorder traversal of a tree
- define a function to find a data item in a tree
- define a function to find the max item in a tree

Part 1: Logging in (choose one person)

1. Follow the procedure from prior labs to log in and open Mobaxterm. Go into your cs305 directory. Make a new `lab8` directory:

```
cd /drives/p/
cd cs305
mkdir lab8
```

2. Get the lab 8 files. Download `tree.zip` from moodle to your `lab8` folder. Go into your lab8 folder and type:

```
ls
```

Hopefully, you can see the file. Unzip them:

```
unzip tree.zip
```

Part 2: Height and Postorder

1. Open `tree.h` to see the struct definitions and the function prototypes.
2. Open `main.c` and look at the function `test1`. What does the tree look like before it is free'ed?
Put drawing here:

3. Compile and run the program. At this point, some of the functions are not returning the proper values, but it should compile.

```
gcc –o runtree tree.c main.c
./runtree
```

4. Open `tree.c`. Scroll to the bottom (you are welcome to look at the code earlier in the file, but it is not necessary). Complete the function definition for `height`. If a tree is empty, the height is 0. Otherwise, the height is 1 + max(height of left subtree, height of right subtree). This function should be written recursively. A max function is defined in the file for you to use.

5. Test your function by compiling and running the code. Does the height match the height of the tree you drew above? ______________

6. Complete the function definition for `postorder`. This should visit the tree in postorder fashion. Use the visit function as a subroutine. You may scroll up higher in the file to see definitions for inorder and preorder to see examples.

7. Test this function by compiling and running the code.

Checkpoint 1 [30 points]: Show your lab instructor/assistant the results of your program running and show them the function definitions for height and postorder. Show your answers to the questions above.

Part 3: Find

1. Now, you will complete the function `find` in `tree.c` . This function should return NULL if the data item d is not found in the tree. If it is found, it should return a TreeNode * (memory address of node) of the node containing the data item d. Recall that the tree is a binary search tree. If the current node's value is equal to d, then return that node. If d < current node's value, then call find on the left subtree. Else, call find on the right subtree.

2. Test your code by compiling and running it.

Checkpoint 2 [20 points]: Show your lab instructor/assistant the results of your program running and show him/her the function definition for find.

Part 4: findMax

1. Complete the `findMax` function. This should return -99999 if the tree is null. Otherwise, it should return the max value found in the tree. The max value in a BST is the right-most node in the tree. If the current node has no right subtree, the current node is the right-most node. If the current node has a right subtree, update the current node to the right subtree. Keep processing until you reach the right-most node and return its value.

2. Test your code by compiling and running it.

Checkpoint 3 [20 points]: Show your lab instructor/assistant the result of your program running and your function definition for findMax.

If you finish early and want more practice, try writing code to:

- sum all the values of nodes in a tree
- count the number of internal nodes with two children

Close Mobaxterm and any other applications. Log off the computer.

If any checkpoints are not finished, they SU NLT midnight. You may submit screenshots, code files, and answers to questions electronically no Moodle or submit printouts.