

Lab 4

5 points

Binary Trees in C or C++

Program 4 involves an insertion sort using trees. Lab 4 is a small step towards the fourth programming assignment. This program can be completed using C or C++. We may not cover C++ in lecture prior to the lab, but since Program 4 has to be done in C++ you may want to use C++ if you have previous experience with C++. The form of the input to lab 4 is similar to lab 3 with one command line argument:

```
./lab4 students
```

and an example is:

```
./lab4 45
```

Assume lab 4 reads in **students** lines of input from a script file where each line of input takes the form:

```
studentID g1 g2 g3
```

where **studentID** is a 9-digit integer student ID and **g1 g2 g3** are student test scores between 0 and 100 on three assignments. The test scores are to be read in as floating point numbers.

For lab 4, you are to create at least two functions to operate on a **binary tree**: **insert**, **print_tree**.

insert

The **insert** function builds and maintains a binary tree in **studentID** order. **insert** takes one student's information and inserts a node for that student in the correct place in the binary tree. Each node should contain the **studentID**, the three test scores and the average test score for that student.

Note – For this lab you need to add some validity checks to the input data. Assume that **studentID**'s are unique and that there should only be one entry per student.

print_tree

The **print_tree** function traverses the binary tree and prints out all the nodes in the binary tree in **studentID** order. Print out all the student information one line per student and include a header line to identify the columns of the output.

Lab 4 main program

The lab 4 program reads in **students** input source lines. For each valid input line, it creates a binary tree node, and inserts this node in the tree. Once the tree is completely built, the program prints out **all** of an individual student's information in **studentID** order from lowest to highest number.

Lab 4 Assignment

0. Prior to coming to the lab prepare a preliminary solution to the program above.
1. Create a make file.
2. Test the program under your own test data input from the terminal.
3. Run the program on the provided test data file 'lab4.dat' redirecting the output to binarytree.txt.
4. Create a README file that contains any useful information to assist in the grading of your lab program.
5. Create a tarred file that contains all the source and header files, the make file and the README file and your output file.
6. Use the Unix version of the 'turnin' to turn-in the tarred file. [The deadline for all lab turn-ins is 24 hours after the beginning of your assigned lab.]