CS163 Final Proficiency Demos

In your proficiency demo, you will be asked to implement a recursive solution for a BST to insert, remove, find, or otherwise modify a node or the data in the data structure. Make sure to practice with the code supplied in the final proficiency demo directory of the cs163 lab system.

For your proficiency demos, there are two phases. You will be coding a BST algorithm to show fluency in working with recursion and binary search trees in C++. You will also be demonstrating navigation, search and replace as well as other linux editor commands. Make sure you are fluent using one of the linux editors (vi, vim, or emacs).

The way the proficiency demos work is as follows:

1. It is a closed book, close notes event. No additional technology (phones, smart watches, etc.) may be used during the demonstration
2. The final proficiency demo is scored within 45-60 minutes after the technology checks (which can take up to 15 minutes to perform)
3. You will be asked to read a document that describes in detail what is expected during the demonstration. It is important to read this to familiarize yourself with the requirements.

4. All applications must be closed prior to coming into the demo if you use your own computer.
5. There must NOT be any code in the clipboard or on the screen. In the event that code is on your screen or clipboard, it will be an automatic failure.
6. After the process of a systems check, the next step will be to select a question, randomly. Read the first question. If you are happy with the question, then you will proceed to the next step. If not, go to the next page and read the next question assigned. Again, if you are happy with this second question, you can proceed. If not, go to the next page. The third question "sticks" and there are "no go backs".

7. Once a question has been selected, it is considered a "closed book, closed notes" session and no further internet access is allowed. **Once you have started working on a question, it cannot be replaced.**

8. The prototype supplied may not be altered. But you may always write more than a single function to solve the problem.

9. **If you have a question during the demo, physically raise your hand and we will assist you as soon as possible.**

10. **Practice Questions:**

    a. Warm up questions (do these first WITHOUT looking at any material).
        i. Add a new item (99) into a BST
        ii. Count the number of nodes in the tree
        iii. Remove all items from a BST
        iv. Display the tree in order

    b. Now, that we are "warmed-up" we can move on to more interesting questions:
        i. Count the number of times a value passed in from the user appears in the tree
        ii. Add a new item (99) into a BST but only if it doesn't already exist in the tree
        iii. Find out if every leaf is at the highest level (e.g., height) of the tree
        iv. Count the number of nodes that have no children
        v. Count the number of nodes greater than the root's data
        vi. Compute the average of all data in the tree
        vii. Make a complete copy of the data in a BST

    c. OK, now let's progress to the next level. Only attempt these after you have done all of the previous questions. Again, these ONLY help if you can do them logically, without looking at materials and without just memorizing answers:

        i. Determine if the root's data appears in the tree more than once
        ii. Copy the contents of a BST and place it in an array, sorted
        iii. Display just the largest data item
        iv. Advanced: Display the largest TWO ITEMS in a BST
        v. Advanced: Make a complete copy of all even data in a BST

11. Once you have entered the testing arena and begin working on the problem assigned, keep the following in mind:
    a. You are encouraged to spend time with **design**; scratch paper is provided in the testing arena

    b. You are also allowed to **compile, test, and debug** your work. Please consider performing incremental implementation, compiling code as you go.

    c. Remember when coding, **please comment out code rather than deleting it**
    d. It is expected that your solution will minimize traversals. This means that if you have a question to remove the last node and return the total number of nodes in the list, then it all needs to be done through one traversal. Do not attempt to traverse to remove and then traverse again to count!
    e. You may write **more than the function** supplied if necessary, but main must call the function assigned
    f. The prototype(s) supplied **may not be changed**

g. For questions that have you count, average, or traverse in some way, your **solution should NOT modify the data structure**
h. For questions that have you insert or copy, your solution should **NOT delete items**
i. **Avoid extra traversals** just to calculate the returned value (for example, if the question asks to count or sum the data, that should be accomplished as part of the recursive problem and not through an extra pass through the data set)
j. **No use** of the **internet** except to login to our testing environment
k. **No use** of **help** or the **man pages** during the demonstration
l. **No** other accounts may be logged into during the demonstration
m. **Only one shell** session may be open

12. Syntax requirements include:
    a. **No global variables** may be used
    b. **No** use of the **string class**
    c. **No** use of **static locals**
    d. All questions will expect a **recursive solution** be used to solve a repetitive problem
    e. **NO LOOPS may be used in your solution**

13. When we have scored your work you will be asked to submit the results using the submit script. Please wait until prompted to do so by the proctor.

14. If you go over time, the system will automatically warn you with a message, submit your work for you and log you out of the system.