CS260 - Programming Assignment 6 - Binary Trees

The purpose of this assignment is to give you some practice working with binary search trees.

Overview

In this assignment, you'll write the logic to create, destroy, and copy binary trees, and insert and delete nodes (some of this was done in class, and some of the rest is the book, so there's been a few things added (function pointer!) to make it at least minorly challenging.

Operational Details

Your program should do the following:

- Create a binary tree of strings. (use strings, not null-terminated character arrays. The string class encapsulates a null-terminated character array and is easier to work with because, among other things, the comparison operators work with it)
- Insert the following names (in the order given here): Kirk, Chekov, Sisko, Chapel, Data, Janeway, Sulu, Bones, Q, Scotty, Uhuru, Picard, Spock, Worf
- Print the tree contents in order. (see technical details)
- Make a copy of the tree
- Delete Bones and reprint the tree (denote who you're deleting; e.g. print: "After deleting Bones: " then print the tree. Do the same for each of the following deletes.
- Delete Sisko and reprint the tree
- Delete Sulu and reprint the tree
- Delete Kirk and reprint the tree
- Print the copy of the tree using inOrder, then postOrder, and lastly preOrder processing, with each using the same print function (passed by a pointer) (see technical details below)

Technical details

- You should implement a binary tree class.
- You may implement it as a template (more flexible) or not (possibly easier, but much less flexible)
- You should have generic inOrder, postOrder, and preOrder processing routines that take pointer to a function that accepts a reference to a string (or DataType if you're creating a template) AND an integer. The integer will be used to pass the level of the node to each recursive function. (Nodes don't know what level they're on, but when traversing the tree we can keep track using an integer that is incremented when passed via the recursive call; it needs to be passed by value) When the names are printed out, print the level of the node containing the name in (). For example, after the first four names are added, printing the list inOrder would show:
    - Chapel(2), Chekov(1), Kirk(0), Sisko(1)

Very Important Stuff

Program should be well written and function properly.

All programs should follow the class's Coding Conventions
Submit the following:

- A zip file containing all your .cpp and .h files and your executable