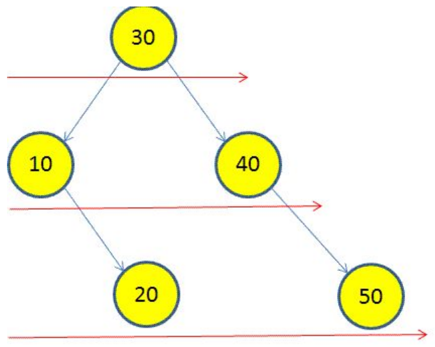
CptS223 Micro Assignment #3 - BST Big Five + Level Order Operations

For this micro assignment, you will be starting with a working BST tree. Your task is to implement the C++11 big five, just as you did for the linked list microassignment. Additionally, you will need to implement a new print function for the tree: printLevelOrderHelper. I also have a function called 'returnLevelOrderHelper' stubbed in. That function returns a vector of the values from the tree in level order. The included source for the BST tree is fully functional up to the Big Five and Level Order stages. The six functions you'll need are well delineated in the file BST.h.

I've stubbed in the Big Five, printLevelOrderHelper, and returnLevelOrderHelper. There are plenty of tests in the BST.h file to cover these in the API. As you complete them, the tests should start outputting the correct data. The Big Five should be very similar to our linked list project, but with fewer values to copy/preserve. The more interesting solution will actually be in printLevelOrderHelper and returnLevelOrderHelper.

Printing a tree in level order means printing each level from left to right, then proceeding down to the next level. This is best done with a queue-based solution instead of the stack based ones we've seen before. Your solution MUST use the C++ STL queue class. The solution is easiest done with a loop instead of recursion. The only difference between the print and return level order functions will be where you send the values (to STDOUT or put the value on the end of the vector).



The code must compile and run on the EECS SIG servers (e.g. sig1.eecs.wsu.edu) using g++. Like usual, I have provided a Makefile to build and test the code. To just build your code, execute 'make'. To run the test suite, run 'make test'. The tests are getting long now, so the first ones will scroll off of the top of the terminal. If you want to learn a great tool to look at and scroll through the output, look up the Linux command "less". I found myself doing this command: 'make test | less'. (pressing 'q' quits less) That will let you look at the output without having to scroll back up the terminal. less is a great tool for looking at and searching through files, too.

There's a 'make bigtest' target in the Makefile. It will generate a tree of 2^20 nodes containing random values from 0..1,000,000. For the bigtest you won't need to do 'make bigtest | less' - it will actually prevent scrolling off of the end during the tree build. If you'd like to see how I accomplish that, it's a neat trick with the \r (carriage return) character.

Grading

Your submission will be graded based on the following:

1. [8] Your modifications cause no runtime issues, your Big Five, printLevelOrderHelper prints out the proper results as shown in the tests when you run make test, and returnLevelOrderHelper's vector results compare correctly with the test vector.
2. [2] Your modifications contain good style. For example,
    - You provide meaningful variable names
    - You provide sufficient and meaningful comments
    - Your code is well structured

Submission Process

The assignment is turned in by submitting the hash of the commit you'd like the grader to check. Make sure to commit your code, push to the git server and test it so you know it's a good version.

FYI: If you telnet to a server, you can end the session by typing ctrl-] (^]), then 'quit'.