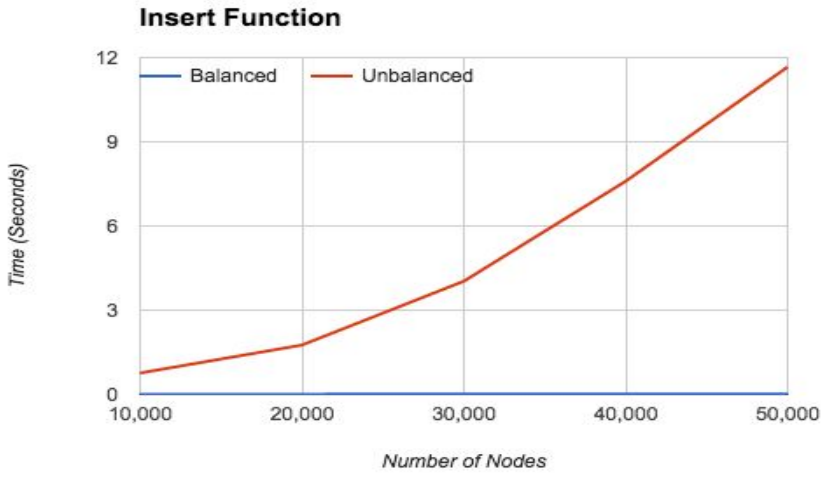
Group: Andrew Lee, James Yao, David Dvorkin

Lab 6

The problem encountered in this lab is discerning the difference between runtimes for a balanced binary search tree and an unbalanced binary search tree. The runtime for the balanced binary search tree should be much shorter than the unbalanced binary search tree with larger input, due to the runtime complexity of the respective trees. An unbalanced binary search tree, considering worst case where it functions much like a linked list, would have a runtime complexity of O(N). A balanced binary search tree, by comparison, would have a runtime complexity of O(logN). Therefore, runtime for a balanced tree should be shorter.

The tests set up check the insert, remove, and find functions for the balanced and unbalanced binary search trees. For the insert function, to place 50,000 nodes, the balanced tree took 0.017 seconds, while the unbalanced tree took 11.665 seconds (seen in the graph below). This is a huge difference, and expected based on the runtime complexities of the trees. Similar results can be seen for remove and find functions.



In conclusion, balanced binary search trees are much faster as opposed to unbalanced trees for insertion, removal, and find due to the logarithmic aspect of their runtime complexity.