Time Complexities:

printInorder(), printPostorder(), searchName(name), and printPreorder() all use helper functions that are O(n) complexity, because we access every node in the tree, and n is the total number of nodes in the tree.

printLevelCount() is constant time as we just access the root node's height, which is 1-based.

searchID(ufid), insert(name, ufid), and remove(ufid) are all O(log n) complexity, because each time we traverse down the tree, the number of nodes we deal with is effectively halved, and n is the height of the tree.

removeInorder(iterations) is O(n) complexity, where n is the integer argument passed into the function.

Reflection:

Overall, I had a fun time coding up this assignment. It was a challenge, but the tree visualization function I built helped a lot towards getting to the solution. I learned how to practically build an AVL tree, encapsulate an API, unit test code to prevent bugs and streamline development, and brushed up on managing memory efficiently. The only thing I would do differently is start earlier!