Lince Rumainum
Student ID: 112722337
CS 2413 Data Structures
Project 4

For Project 4, the time complexity of inserting n objects into AVL tree while checking for duplicates is *O(log n)*. Since the AVL tree will always be balanced (rebalanced itself when it is not, i.e. height > 1 or height < -1) with any data greater than its root on the right side and data less than its root on its left side, the data in the tree is being sorted while it is being inserted. Therefore, the time complexity is *O(log n)* because as it is trying to insert the item in a sorted tree, it will check if the data in that node is a duplicate or not. On the other hand, data in a sorted linked list will have a time complexity of *O(n).* Although insertion in linked list is *O(1)*, since it will go from the beginning of the list and compare through each node until it finds the place where it should be inserted or find its duplicate, the time complexity becomes *O(n)*.

The time complexity for the merge algorithm for the tree will be *O(n+m)* since both trees are in-order and will be merge together in parallel. It will go through both list and inserted for whichever one have the smaller record ID value, retain the new read in data when there the data is a duplicate, and inserted the rest of the tree that still have data left after merging both trees.

The space complexity for the merge algorithm will also be *O(n+m)* as long as there is no duplicate in the data. The new tree will have to hold all the data from both trees, which has *n* and *m* size, after they are being merge.

The time complexity of the purge algorithm for the tree will be *O(n log m)* since *n* is the size of the tree containing the file data and *m* is the size of the initial tree where data will be removed. It is a *O(n log m)* operation because the program will go through each data of the file data, which will be *n* times, and for those data go through the initial tree to remove data, which is a *O(log n)* operation.

The space complexity of the purge algorithm for the tree would be *O(m)* in which case that none of the data from the initial file are in the purging file so the AVL tree will be kept as it was at initial reading.