AVL TREE

Ahmed Ibrahim

Trees are considered more efficient that some data structures due to its average complexity of O (log n) in search, delete, and insert. However, it has a worst case of O (n), which makes it no different than a Linked List or an Array. To overcome such issue, the implementation of an AVL Tree was introduced to balance the tree and have the average complexity of O (log n) the main case. In this program, we used AVL Rotations to make the tree balanced in height. To do so, an implementation of "updateHeightandEvaluateBalance" was created to check for the balance factors for every insertion in such tree, then updating the whole tree until reaching the root. Even though such operation had a complexity of O(n), it reduced the other operations like, search, delete ,etc., to have a complexity of O(log n). Such complexity would be trivial in comparison with the speed such tree can have with large data.