COP3530 – Data Structures and Algorithms
Project 1
10/09/2023
Benjamin Uppgard

Method Time Complexity (Worst Case)

Note: Unless otherwise stated n is the number of nodes in the AVL data structure.

> `bool InsertStudent(unsigned int studentID, string& studentName)` – O(n)) – the properties of an AVL tree are leveraged to minimize the number of nodes that must be visited to determine the correct insertion location, however if the node is inserted at the root of the tree, each node must be visited to calculate the new balance factor.
>
> `bool Remove(unsigned int studentID)` - O(n) – the properties of an AVL tree are leveraged to minimize the number of nodes that must be visited to find the node that will be deleted, however if the root node is removed each node in the tree must be visited to update the balance factor
>
> `bool SearchID(unsigned int studentID)` - O(log(n)) – the properties of an AVL tree are leveraged to minimize the number of nodes that must be visited to find the desired node.
>
> `bool SearchName(string studentName)` - O(log(n)) – the properties of an AVL tree are leveraged to minimize the number of nodes that must be visited to find the desired node.
>
> `void PrintInOrder()` - `void PrintPreOrder()` - `void PrintPostOrder()` – O(n) – each of these functions must visit each node in the tree once.
>
> `void PrintLevelCount()` – O(n) – each node must be visited to find the maximum depth from the root node to the left and right.
>
> `bool RemoveInOrder(unsigned int n)` - O(n) – in the worst case scenario, the input parameter n is larger than the number of nodes in the AVL data structure. In this case, all nodes must be visited.

Looking Back: Recursive functions have been a difficult concept for me to wrap my head around. Working with the AVL data structure has helped me to have a more intuitive understanding of these functions. I still think that I have some improving to do with that. If I were to start this project again, I think that I could make the code of my functions a lot simpler and cleaner. I didn't realize that my function for determining the balance factor was O(n) until too late in the project. If I was able to do things again, I would optimize the data structure to update the balance factor more efficiently during insertion and removal. The challenges that I faced with the recursive functions also delayed my experimenting with test cases. Test cases are also new to me, but If I were to start this project from the beginning, I would have implemented and utilized unit testing from the very beginning.