CS499: Milestone Three: Data Structures and Algorithms

Antonio Arruda

Southern New Hampshire University

**Artifact**

This artifact was created during my Data Structures and Algorithms course here at SNHU. During the course, we created a program to search CSV files to insert, remove, and search for items sold through a website. Our program needed to be user-friendly and efficient. In order to gain efficiency, we used binary search trees, vectors, and hash tables to traverse the data structures and explore the data provided.

This Milestone is centered around the Data Structures and Algorithms expertise. To show my understanding of data structures and algorithms, I wanted to find a way to make my binary search tree more efficient. After some research, I learned about the AVL binary search tree, or AVL Tree. The AVL tree is logically more efficient than the binary search tree as it is self-balancing meaning the difference in the node’s height can be no more than plus or minus one. Using functions to rotate the binary tree to shorten its height, we can improve the efficiency of the binary tree. For example, binary search trees are calculated by O(h) where h is the height of the search tree. The AVL search tree is calculated by O(Log n) where the time increases linearly as the number of nodes increases exponentially. Though the time complexity is the same, the AVL tree prevents the height of the tree to be skewed and speeds up the process.

I think that this meets the criteria for this milestone as it accomplishes the plan I had originally outlined in Module 1. I set out to increase the efficiency of the binary search tree and converting the Binary Tree to the AVL Search tree accomplishes that. During the process of enhancing the artifact, I learned quite a lot. I found that google was an excellent resource in discovering new algorithms to improve/change the code to be more efficient. I believe the most challenging aspect to this challenge was to figure out how to implement the functions to properly execute the algorithms and result in a functioning AVL tree.