**6-3 Project One: List Evaluation and Recommendation**

Analyzing the Big O equations for each list type can help determine which would be best for ABCU. I will review some functions and show the O equation used.

Vector

Sort: Worst O(N^2) – Best O(N)

Resize: O(N)

Search: O(N)

Insert/Remove: Worst O(N) – Best O(1)

Hash Table

Sort: O(N)

Resize: O(N)

Search: O(1)

Insert/Remove: Worst O(N) – Best O(1)

Binary Tree

Sort: O(N)

Resize: O(1)

Search: Worst O(N) – Best O(logN)

Insert/Remove: O(1)

The analysis shows how each function is done using a linear, logarithmic, or constant approach to using different tasks with the amount of data. To simplify the efficiency of each equation, O(N) (linear) means it goes over each node till a specific node is identified, O(1) (constant) means it goes straight to the intended node, and O(logN) (logarithmic) splits amount of nodes in half till it reaches the target node (32, 16, 8, 4, 2, 1). Each part of a list has varying degrees of speed based on its task. To find the best list type, investigating ABCU’s needs will help make the right decision.

The program they request must load courses, print the entire list sorted in alphanumeric order, and print out a single course searched for. By putting a value of two for O(logN), three for O(1), and one for O(N), I can determine which will be the best list for ABCU. I will also apply the best-case scenario. Loading courses can be divided into three processes: sorting, resizing, and inserting.

Vector: 1+1+3 = 5

Hash Table: 1+1+3 = 5

Binary Tree: 1+3+3 = 7

Next, a sorted list is printed, which is searched, inserted, and sorted.

Vector: 1+3+1 = 5

Hash Table: 3+3+1 = 7

Binary Tree: 2+3+1 = 6

Finally, it searches for and prints a specific node so it can be labeled as a search.

Vector: 1

Hash Table: 3

Binary Tree: 2

Now, tally the points to find the best option.

Vector: 5+5+1 = 11

Hash Table: 5+7+3 = 15

Binary Tree: 7+6+2 = 15

The results show that a Hash Table or Binary Tree is the optimal option based on the criteria needed to run an efficient program for ABCU. The main difference is that searching is faster using a constant approach from a hash table, whereas using a logarithmic search can be just as fast. However, increasing the tree's height over time will slow the search. Therefore, I would have to recommend the Hash Table because, should the size of files increase, the Binary Tree will slow down over time.