Hadi Abdelaal

CS-300

Dr. Ling

**The Runtime Analysis and Memory for data structures:**

A white rectangular box with black text

Description automatically generated

A line cost sheet with black text

Description automatically generated

A table with numbers and text

Description automatically generated

A line cost calculation table

Description automatically generated

A table with text and numbers

Description automatically generated with medium confidence

Every information format has benefits and downsides for the application's demands. The vector technique has the benefit of offering the quickest way to access the file and add the necessary objects. It is a very simple approach in which each item is merely added to the end of a vector when the file is processed. The overall duration of the three approaches was the smallest, at 5n+1, despite the fact that they all used the same O(n) syntax. One downside of utilizing a vector is that it is difficult to narrow down the list for a certain course. The computer has to search over everything in the direction vector before it finds one that matches.

Hash tables provide the benefit that you're able to swiftly explore a list. The precise location of a particular path will be recognized by establishing a key, which can be readily accessed and published. It is a slower method when constructing the first list since a key need to be defined for every item and a location identified to put every course. Furthermore, hash tables are unsuitable for sorting. The data table is unable to be searched on its own. Every value has to be retrieved, organized, and printed in order to generate an alphabetic list of all classes. This implies that it is most likely not the optimum data format for this particular application.

Binary trees offer the benefit of being quicker when searching than vectors. It is fairly simple to run along the tree once the significance is discovered if you're familiar with the course being sought. It may not be as simple as a hash table; however, it is faster than a vector. If the tree finished with its left departs it would have needed to be looking for each component. The amount of search time would thus be O(h), wherein h is the peak of the tree.

Finally, I would suggest vector sorting for this particular endeavor. I believe that having the ability able to quickly sort and print the whole collection is beneficial to the consumer. Furthermore, the absence of time for searching is not as severe as an advantage of the sort. In general, I believe the vector is the greatest choice, based on the choices and the pros of using vectors as I mentioned above.

**Resources:**

1. *Trees In C++: Basic Terminology, Traversal Techniques & C++ Tree Types*. (n.d.). Www.softwaretestinghelp.com. https://www.softwaretestinghelp.com/trees-in-cpp/