Menu for courses

Print Enter Choice:

Cin >choice

Switch(choice)

Case 1 courses = load structure; break;

Case 2 = print courses(courses); break;

Case 3 search course(courses); break;

Case 4: exit; break

Default: invalid choice

While ch!= 4

Return 0

Function to print courses in order

Void printcourses(vector<course>)

N = courses.size()

For loop to iterate through vector i++

For loop to iterate again j++

If courses[j] > courses[i]

Swap j and i

For loop to print i++

printCourses(courses[i])

To analyze the time to run using our pseudocode, we must look at the cost per line and how many times the line will run. When reading the file it will need to run 1 time, the read each line in the file it will run 1 time per course, and 1 cost to close the file. So the total cost is 2 + total number of lines in the file. To create the course object it will 1 to parse the line and 1 to create the object + however many times the action is executed. In order to know how much it will cost to add to the data structure, we first must know what structure we are using.

For a vector, the worst-case runtime is O(1), and if it must be resized it is O(n). Advantages for a vector are that elements are easy to access and adding new elements is efficient. Disadvantages are that resizing can be inefficient and changing elements in the middle of a vector is slow.

For has tables, the worst case is also O(1), assuming the function distributes keys evenly. If all keys hash to the same value, worst case is O(n). Advantages are that insertion and retrieval of elements is generally fast and the size is easily adjustable. Disadvantages are that collisions can make has tables slow, and the order is not preserved.

Tree’s have a worst case of O(log n), but if it’s unbalanced then worst case is O(n). Advantages are that inserting retrieving and deleting elements is fast, and the order is preserved, disadvantages are that the run time can skyrocket if it becomes unbalanced, and it requires far more memory than other data structures.

For me, I will use vectors in order to create this program. Vectors are easily accessible, and adding new elements will be easy. The main use of this program is to print out course info, so once the vector is created, it won’t be changing too much. The upfront runtime is a good tradeoff for the accessibility that we will get when executing the main part of our code.