Justin Lennox

10Apr2025

CS-300 DSA: Design and Analysis

Professor Rissover

Project One

**Pseudocode for the Menu.**

Function Menu()

WHILE True

PRINT "Select an option:"

PRINT "1: Load courses into data structure"

PRINT "2: Print all courses in alphanumeric order"

PRINT "3: Print course title and prerequisites"

PRINT "9: Exit program"

Input option

IF option == 1

CALL StoreCoursesInTree(fileName)

PRINT "Courses successfully loaded."

ELSE IF option == 2

Call PrintCourseInformation(coursesTree.root)

ELSE IF option == 3

PRINT "Enter course number:"

Input courseNumber

Search coursesTree for courseNumber

IF course exists

Call CheckPrerequisites(course, coursesTree)

ELSE

PRINT "Error! Course not found."

END IF

ELSE IF option == 9

PRINT "Exiting program."

Exit While

ELSE

PRINT "Invalid option. Please select again."

END IF

End While

End Function

**Pseudocode that will print out the list of the courses in the Computer Science program in alphanumeric order.**

Function PrintCourseList(treeNode)

IF treeNode is not null

Call PrintCourseList(treeNode.left)

PRINT treeNode.courseNumber + ": " + treeNode.courseTitle

Call PrintCourseList(treeNode.right)

END IF

END Function

Function PrintAllCourses(coursesTree)

IF coursesTree.root is null

PRINT "No courses available."

RETURN

END IF

PRINT "Courses in alphanumeric order:"

Call PrintCourseList(coursesTree.root)

END Function

**Evaluation and Recommendation**

While evaluating the different data structures to run this program, I can see different advantages and disadvantages to each. When reading the file, the vector, hash table and binary search tree are all O(n) for the worst-case running time. The same goes for when they are creating a course object. The worst-case running time for all of them is O(n).

When choosing the print option, the cost each line in all three data structures are effective but the binary search tree is the best option. The cost per line for the Vector and the hash table is O(n\log n) which is an efficient speed but the binary search tree has an in-order traversal efficiency of O(n). The reason that the binary search tree is the most efficient is because it maintains the order of information. There is no need for additional sorting.

Based on the requirements, all the data structures have advantages and disadvantages to them. The vector is easy to implement. If you do not have much data and you know you will not be adding too much data then the vector is the best option for small datasets. Its downside is that it uses a linear search which can be slow. If the item that you want is at the end of the vector or not in the vector at all then it will take a long time to search through the entire vector. When sorting the data, a vector uses up time to sort the data completely.

A hash table is a very good option when it comes to the courses. It is very fast when looking up and inserting new courses. It is also very good at looking for specific courses and finding their prerequisites. The downside is that sorting the keys for a hash table adds an extra step and the performance depends on the quality of the hash function. If there are many collisions, it will degrade the performance.

A binary search tree is also a good option for a school with several courses and prerequisites. It naturally keeps all of the courses in alphanumeric order by inserting them where they should be from highest to lowest and left to right. The downside is that when the tree becomes unbalanced it will slow down the insertion and search to O(n) which is still not a terribly inefficient speed. They are also more complex to create compared to vectors and hash tables. Removals and insertions while trying to keep it balanced can prove to be very difficult.

After analyzing all three data structures, I recommend that ABCU use a balanced binary search tree. It is one of the most efficient ways to search for data as it is already sorted and even though it is more difficult to implement, I think it is the best choice for what ABCU wants. It is efficient when looking up course information by having all the courses sorted preemptively.