Jodie Snodgrass

11 June 2025

CS 300

Project 1

**Menu Pseudo Code**

WHILE int user choice does not equal 9

PRINT “Menu Options: “ end line

PRINT “1. Load File” end line

PRINT “2. Print All Courses” end line

PRINT “3. Print Details of Course” end line

PRINT “9. Exit Menu”

IF user input is 1

LOAD data from files

IF user input is 2

PRINT all courses

IF user input is 3

PRINT “Course ID: “

User INT courseNumber

PRINT “Course ID” <courseNumber> <courseName> “ Prerequisites required” <prerequisites>

IF user input is 9

PRINT “Good bye.”

EXIT menu

**Print Order Pseudo Code**

**Vector**

INITIALIZE empty vector

FOR each course

ADD course to vector

END

SORT vector in alphanumeric order

FOR each course in vector

PRINT course

END

**Hash Table**

INITIALIZE empty hash table

FOR each course

ADD course to table with courseNumber as the key and course as the value

END

INITIALIZE empty list

FOR each key in table

ADD key to list

END

SORT list in alphanumeric order

FOR each key in list

PRINT table[key]

END

**Binary Tree**

INITIALIZE empty binary search tree

FOR each course

INSERT course into tree with courseNumber as key and course as value

END

FOR each course in INORDER traversal of tree

PRINT course

END

**Vector**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operations** | **Line cost** | **Times executed** | **Total Cost** |
| CREATE vector | 1 | 1 | 1 |
| OPEN and read file | 1 | n | n |
| Parse each line | 1 | n | n |
| ADD course and prerequisites to vector | 1 | n | n |
| PRINT vector course | 1 | n | n |
| **Total Cost** | 5 |  |  |
|  |  |  | 5n+1 |
| **Runtime** |  |  | O(n2) |

**Hash Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operations** | **Line Cost** | **Times Executed** | **Total Cost** |
| CREATE hash table | 1 | 1 | 1 |
| OPEN and read file | 1 | n | n |
| Parse each line | 1 | n | n |
| FOR all courses | 1 | n | n |
| FOR all prerequisites | 1 | n | n |
| ADD course and prerequisites to table | 1 | n | n |
| PRINT table | 1 | 1 | 1 |
| **Total Cost** | 7 |  | 7n+2 |
| **Runtime** |  |  | O(n) |

**Binary Search Tree**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operations** | **Line Cost** | **Times Executed** | **Total Cost** |
| CREATE binary tree | 1 | 1 | 1 |
| OPEN and read file | 1 | n | n |
| Parse each line | 1 | n | N |
| FOR all courses | 1 | n | N |
| FOR required prerequisites | 1 | n | N |
| INSERT course to tree | 1 | n | N |
| GET course | 1 | n | n |
| PRINT searched course and prerequisites | 1 | n | n |
| **Total Cost** | 8 |  | 8n+1 |
| **Runtime** |  |  | O(n) |

**Advantages and Disadvantages**

The first milestone was designed for a vector. There are some advantages to vector structures which include quick file reading, ability to add to the structure, and dynamic sizing. With quick file reading the vector can open the file quickly and read through the list of courses. Having the ability to add to the structure can help if needing to add course objects. The ability to have dynamic sizing can be helpful if adding to the vector list can easily increase in size. While there are great advantages, one disadvantage is it could be a slow search. With a slow search ability as we can see from the tables it may take longer for the user to see their searched results.

The second milestone was designed for a hash table. The advantages of the hash table include fast searches, heterogeneous data, and a reduced space complexity. With fast searches the user can quickly see what they searched including the course number, name, and prerequisites. Heterogeneous data would be beneficial if needing to contain different data types, structures, formats, or sources. With reduced space complexity the hash table can have performance improvements and scalability. There are a few disadvantages to the hash table which include lack of inherent order, hash collisions, and memory usage (Gupta, 2024). With a lack of inherent order the hash table would not maintain sorted data. Hash collisions may occur if different keys produce the same value which can affect performance. Memory usage can be a disadvantage to the hash table as it may be required to avoid or minimize collisions.

The third milestone was designed for a binary search tree. The binary search tree method has some advantages which include efficient searching or sorting and being memory efficient (Gupta, 2024). Having the ability to efficiently search can the user easily and quickly find what they have searched for. With efficient sorting this would be helpful when sorting to order to display courses. The ability to be memory efficient could be helpful only reserving the memory of current elements free space and increasing performance overall. The disadvantages of a binary search tree include modification time and performance sensitivity. The disadvantage of modification time is if inserting or deleting elements can be slow. Performance sensitivity is a disadvantage as performance of the system can be affected by the order of input data.

**Recommendation**

Reading over the advantages and disadvantages of each method along with the timing chart was very helpful to decide what method would be best. Considering the hash table with have difficultly maintaining a sorted order I knew that would not be best. That consideration left me to decide between vector and binary search tree. While the binary search tree could work, I believe a vector would be best for this project. The vector method would be able to open, read, and sort data quickly would make up for the slower search ability. With its ability to increase in size as needed is also beneficial incase any courses were to be added.

References

Gupta, Himanshu. (2024, July 30). Advantages of BST over Hash Table. GeeksforGeeks.

https://www.geeksforgeeks.org/dsa/advantages-of-bst-over-hash-table/.