Mikayla Grady

Professor Andujar

CS-300: DSA: Analysis and Design

20 February 2022

Project One

**Pseudocode for Menu:**  
//Creating Menu  
//Menu needs to display upon entering program

Initialize program  
//Viewing Main Menu Selections  
displayMenu   
 “Main Menu” displays

“1. Load Data” displays  
 “2. All CS Courses” displays  
 “3. Individual Course with Prerequisites” displays  
 “4. End Program” displays  
//User enters selection determining next steps

If user selects “1”

Then data from file loads  
If user selects “2”  
 Then all courses display  
If user selects “3”  
 Then “Enter Course ID” displays  
If user inputs course ID  
 Then input course and prerequisites display   
Else if user enters invalid ID  
 Then error “Invalid Entry” displays  
 Then user is returned to homepage  
If user selects “4”  
 Then “Goodbye” displays   
 Then program ends

**Pseudocode for Course list:**

Binary Source Tree:   
// Pseudocode for Course list  
// Printing course and all needed prerequisites  
Initialize

//Define constructor  
//Print  
void printCourseInformation   
if node != nullptr

Then print elements

If node->

**Run-Time Evaluation:**

Binary Source Tree (Print):

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| Initialize | N/A | N/A | N/A |
| void printCourseInformation | 1 | 1 | 1 |
| node != nullptr | 1 | 1 | 1 |
| Then print elements | 1 | n/2 | n/2 |
|  |  |  |  |
| **Total Cost** | | | 4n + 1 |
| **Runtime** | | | O(n) |

**Advantages and Disadvantages:**

Each of the above data structures (vector, hash table, and binary search tree) all have unique advantages and disadvantages.

Vectors are fairly easy to implement however, they do take up significantly more memory. Also, in terms of speed, Vectors are not the best option out of the three.

Some benefits of Hash Tables are that they’re good for large quantities of data and are also extremely quick and efficient. However, Hash Tables also have some downfalls such as collisions that occur and can potentially cause problems.

Binary Search Trees are another great option that is extremely quick and efficient. However, Binary Search Trees are also considerably harder to maintain and things such as the balance needs to be considered. Also, it is important to remember that in order for Binary Search Trees to work, the list has to be sorted.

**Recommendation:**

For this specific project, I would recommend implementing a binary search tree. There is not an unreasonably large amount of data, so the access speed provided from a binary search tree is acceptable for this application. This application also requires the courses to be in alphanumeric order. A binary search tree is better equipped to show the courses in alphabetical order than either of the other options (vector or hash table). Because of the scope of this application, it should be fairly easy to maintain a balanced tree or make sure it is self-balancing.