6-2: Project One

Jorgo Qendro

Southern New Hampshire University

CS-300: Analysis and Design

Professor Adamo

October 12th, 2024

6-2 Project One

# **Pseudocode for File Handling**

function loadFile(String filename) {  
 open file  
 while not end of file {  
 read line  
 parse line into courseNumber, courseTitle, prerequisites  
 if format is valid {  
 create a Course object with courseNumber, courseTitle, prerequisites  
 add the Course object to the data structure (e.g., vector, hash table, tree)  
 } else {  
 print "Error: Invalid format in line"  
 }  
 }  
 close file  
 }

# **Course Object Creation**

function createCourse(String courseNumber, String courseTitle, List<String> prerequisites) {  
 Course newCourse  
 newCourse.courseNumber = courseNumber  
 newCourse.courseTitle = courseTitle  
 newCourse.prerequisites = prerequisites  
 return newCourse  
 }

# **Pseudocode for Printing Course Info**

function searchCourse(Tree<Course> courses, String courseNumber) {  
 if tree is not empty {  
 course = search in tree for courseNumber  
 if course is found {  
 print course title and course number  
 for each prerequisite in course.prerequisites {  
 print prerequisite course info  
 }  
 } else {  
 print "Course not found"  
 }  
 }  
 }

# **Pseudocode for Menu Options**

function displayMenu() {  
 print "1: Load course data"  
 print "2: Print list of all courses"  
 print "3: Print course and prerequisites"  
 print "9: Exit"  
   
 get user input  
 if input is 1 {  
 call loadFile function  
 } else if input is 2 {  
 call printAllCourses function  
 } else if input is 3 {  
 call searchCourse function  
 } else if input is 9 {  
 exit program  
 }  
 }

# **Sorting and Printing Course List**

function printAllCourses(Vector<Course> courses) {  
 sort courses by courseNumber  
 for each course in courses {  
 print course number and title  
 }  
 }

# **Runtime Analysis**

| Operation | Vector | Hash Table | Binary Search Tree |  
 |------------------------------|---------|------------|--------------------|  
 | Load File | O(n) | O(n) | O(n) |  
 | Search Course by Number | O(n) | O(1) | O(log n) |  
 | Sort and Print Courses | O(n log n) | O(n) | O(n) (in-order traversal) |

# **Recommendation**

Based on the runtime analysis, I recommend using the Binary Search Tree for this program. It provides efficient search functionality (O(log n)) and is inherently sorted, making it optimal for the alphanumeric course listing requirement. While the Hash Table offers faster search times, the Binary Search Tree's sorting capability makes it more suitable for the project.