Israel Stillman

Michael Susalla

CS-300 23EW1

October 8, 2023

Project One

**Define a function to open and read the file**

try

file = open(fileName, 'r')

for line in file

Call function to parse and store data

close(file)

except FileError

print("Error: Unable to open or read the file")

**Define a function to parse and store data from a single line**

tokens = split(line, ',')

Check for correct number of parameters

courseNumber = tokens[0]

courseTitle = tokens[1]

prerequisites = tokens[2:] # If any

Check if prerequisites exist in courseVector

for prereq in prerequisites

if not courseExists(prereq)

print("Error: Prerequisite course does not exist")

return

Create a course object and store it in courseVector

course = createCourseObject(courseNumber, courseTitle, prerequisites)

courseVector.append(course)

**Define a function to create a course object**

course = new CourseObject

course.courseNumber = courseNumber

course.courseTitle = courseTitle

course.prerequisites = prerequisites

return course

**Define a function to check if a course exists in courseVector**

for course in courseVector

if course.courseNumber == courseNumber

return true

return false

**Define a function to search and print course information**

found = false

for course in courseVector

if course.courseNumber == courseNumber

print("Course Number:", course.courseNumber)

print("Course Title:", course.courseTitle)

print("Prerequisites:")

for prereq in course.prerequisites

print(prereq)

found = true

break

if not found

print("Course not found")

**Pseudocode for Reading and Processing Course Information File:**

Initialize a hash table to store course objects.

Open the Course Information file.

If the file cannot be opened, display an error message and exit.

While there are lines in the file, do the following:

Read a line from the file.

Split the line into tokens (courseNumber, title, prerequisites).

If the number of tokens is less than 2, display an error message and skip this line.

Create a new course object and set its attributes using the tokens.

Add the course object to the hash table using courseNumber as the key.

End loop.

**Pseudocode for Validating Prerequisites:**

For each course object in the hash table, do the following:

For each prerequisite in the course object:

If there is no course object in the hash table with the same courseNumber as the prerequisite,

display an error message indicating missing prerequisite information.

End loop.

**Pseudocode for Printing Course Information:**

For each course object in the hash table, do the following:

Print courseNumber, title, and prerequisites (if any).

End loop.

**Open and read the file:**

If File is successfully opened then

Read each line from the file

Close the file

else

Display an error message and exit

End If

**Check file format and prerequisites:**

for each line in the file do

Split the line into tokens (courseNumber, title, prerequisites)

if number of tokens < 2 then

Display an error message for incorrect format

Skip to the next line

if prerequisites are not empty then

for each prerequisite in prerequisites do

if prerequisite does not exist as a course in the file then

Display an error message for missing prerequisite

Skip to the next line

end if

end for

**Create and store objects:**

Define a class Course with attributes: courseNumber, title, prerequisites

Create an empty vector data structure called courseObjects

for each line in the file do

Split the line into tokens (courseNumber, title, prerequisites)

Create a new course object with courseNumber, title, and prerequisites

Add the course object to the courseObjects vector

end for

**Print course information:**

for each courseObject in courseObjects do

Print courseObject.courseNumber, courseObject.title

if courseObject.prerequisites is not empty then

Print "Prerequisites: "

for each prerequisite in courseObject.prerequisites do

Print prerequisite

end for

end if

end for

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Line Cost | Number of Times Executed | Total Cost |
| Open and read the file | 1 | 1 | 1 |
| Check file format and prerequisites | 1 | n | n |
| Create and store objects | 1 | n | n |
| Print course information | 1 | n | n |
| Total Cost | | | 3n + 1 |
| Runtime | | | O(n) |

**Advantages and Disadvantages:**

* **Vector:**
  + Advantages:
    - Simple to implement.
    - Provides direct access to elements.
  + Disadvantages:
    - Insertion and deletion operations can be slow for large datasets.
* **Hash Table:**
  + Advantages:
    - Fast access to elements through hashing.
    - Good for retrieval of specific elements.
  + Disadvantages:
    - May have collisions, which need to be handled.
    - Hash function quality affects performance.
* **Tree:**
  + Advantages:
    - Efficient for searching, insertion, and deletion (if balanced).
    - Maintains sorted order.
  + Disadvantages:
    - Balancing can be complex and time-consuming.
    - May degrade to O(n) for unbalanced trees.