Stephen Zawatski  
Southern New Hampshire University  
CS-300: Data Structures and Algorithms  
Instructor: Saba Jamaliannasrabadi  
Date: 02/02/2025

**Pseudocode for Course Management System Using Hash Tables**

**Main Function (Menu Loop)**

csharp

CopyEdit

Function Main():

Read command-line arguments

Store argument as CSV file path

IF no command-line arguments are given:

Load default CSV file path

WHILE userChoice is not ‘9’:

Display menu options

Get user input and store in menuChoice

Get user input for data structure type (store in dataChoice)

Validate user input

IF user input is invalid (not 1-4 or 9):

Display error message

IF menuChoice equals ‘1’: // Load Courses

IF dataChoice is HashTable:

Call loadCourses() and store data in hashTable

IF menuChoice equals ‘2’: // Validate Data

IF dataChoice is HashTable:

Call validateHashTable() passing hashTable

IF menuChoice equals ‘3’: // Search for a Course

Get user input for courseNumber

IF dataChoice is HashTable:

Call searchCourse() passing hashTable and courseNumber

IF menuChoice equals ‘4’: // Display All Courses

IF dataChoice is HashTable:

Call displayCourses() passing hashTable

EXIT program

**Function to Load Course Data from CSV File**

sql

CopyEdit

Function loadCourses(fileName):

Open file fileName

IF file cannot be opened:

PRINT "Error: Unable to open file"

RETURN

Initialize empty hashTable

FOR each line in file:

Split line using ',' delimiter

IF line does not contain at least two elements:

PRINT "Error: Invalid course format"

CONTINUE

Extract courseNumber from first element

Extract courseTitle from second element

Extract prerequisites from remaining elements (if any)

Create new Course object

Assign courseNumber, courseTitle, and prerequisites to Course

Insert Course object into hashTable using courseNumber as key

Close file

RETURN hashTable

**Function to Validate Course Prerequisites**

sql

CopyEdit

Function validateHashTable(hashTable):

FOR each course in hashTable:

FOR each prerequisite in course.prerequisites:

IF prerequisite does not exist in hashTable:

PRINT "Error: Prerequisite", prerequisite, "not found for course", course.courseNumber

RETURN False

RETURN True

**Function to Search for a Specific Course**

scss

CopyEdit

Function searchCourse(hashTable, courseNumber):

IF courseNumber is not found in hashTable:

PRINT "Error: Course", courseNumber, "not found"

RETURN

course = hashTable[courseNumber]

PRINT "Course Number:", course.courseNumber

PRINT "Course Title:", course.courseTitle

PRINT "Prerequisites:", formatPrerequisites(course.prerequisites)

**Function to Display All Courses**

scss

CopyEdit

Function displayCourses(hashTable):

PRINT "Course Number | Course Title | Prerequisites"

PRINT "-------------------------------------------"

FOR each course in hashTable:

PRINT course.courseNumber, "|", course.courseTitle, "|", formatPrerequisites(course.prerequisites)

**Helper Function to Format Prerequisites**

vbnet

CopyEdit

Function formatPrerequisites(prerequisites):

IF prerequisites list is empty:

RETURN "None"

ELSE:

RETURN prerequisites joined by ', '

**Struct for Course Object**

cpp

CopyEdit

Struct Course:

courseID

courseTitle

prerequisites (List)

Constructor Course():

courseID = ""

courseTitle = ""

prerequisites = []

**Class for Hash Table Implementation**

markdown

CopyEdit

Class HashTable:

- Struct Node:

Course data

Key (courseID)

Next pointer

+ hashFunction()

+ insertCourse()

+ searchCourse()

+ displayCourses()

**Runtime Complexity Analysis**

| **Operation** | **Hash Table Complexity** |
| --- | --- |
| Loading Data | O(1) - O(N) *(depends on collisions)* |
| Search | O(1) - O(N) *(depends on collisions)* |
| Sort/Print | O(N) |

**Conclusion and Recommendations**

* **Advantages of Using a Hash Table:**
  + Provides **fast lookups** when retrieving course details.
  + **Efficient storage** and quick access to course data.
  + Can handle **prerequisite validation** effectively.
* **Recommendation:**
  + If **searching courses frequently**, a **hash table is preferable** because of its **O(1) average lookup time**.
  + To optimize performance, the **hash function and table size should be adjusted** to reduce **collisions**.