# CS 300 Pseudocode Document

Gentian Hoxha

[Gentian.hoxha@snhu.edu](mailto:Gentian.hoxha@snhu.edu)

4/7/2024

Module 4 Milestone

***Pseudocode***

***Loading Data into a Hash Table***

Procedure readCoursesFromFilePath(filePath):

Open the file at filePath

Create an empty hashTable for courses

While the file has more lines:

line = Fetch the next line from the file

Divide line into courseNumber, title, and prerequisites using a separator (like a comma)

If neither courseNumber nor title is missing:

Initialize a Course object with courseNumber and title

If there are prerequisites listed:

Iterate over each prerequisiteId in the list of prerequisites:

Insert prerequisiteId into the Course object's list of prerequisites

Place the Course object into hashTable using courseNumber as its identifier

Else:

Display "Error: Course number or title not provided."

Close the file

Give back hashTable

***Printing Course Information from a Hash Table***

Procedure showCourseDetails(coursesHashTable, courseNumber):

Check if courseNumber is found in coursesHashTable:

Access the course by courseNumber from coursesHashTable

Output "Course Number: ", followed by course.courseNumber

Output "Title: ", followed by course.title

Output "List of Prerequisites:"

If the course includes prerequisites:

Loop through each prerequisiteId in the course's prerequisites:

Verify if prerequisiteId is in coursesHashTable:

Access the prerequisite course by prerequisiteId from coursesHashTable

Output its courseNumber and title

Else:

Output "Data missing for prerequisite ", prerequisiteId

Else:

Output "There are no prerequisites."

Else:

Output "No course found with number ", courseNumber

***Counting the Number of Prerequisites for a Course***

Procedure calculatePrerequisites(coursesHashTable, courseNumber):

Confirm if courseNumber is within coursesHashTable:

Fetch the course by its courseNumber from coursesHashTable

Set prerequisitesCount to zero

Loop through each prerequisiteId in the course's prerequisites list:

Check if prerequisiteId is present in coursesHashTable:

Increment prerequisitesCount by one

// You might consider counting deeper-level prerequisites

Display "Total prerequisites for ", courseNumber, ": ", prerequisitesCount

Else:

Display "No course found with the number ", courseNumber

***Printing a Sample Schedule***

Procedure outputSampleSchedule(coursesHashTable):

Iterate over each courseNumber and course in coursesHashTable:

Output "Course: ", followed by course.title, " (", and course.courseNumber, ")"

Output "List of Prerequisites: "

If the course has listed prerequisites:

Loop through each prerequisiteId in the course's prerequisites:

Confirm if prerequisiteId is in coursesHashTable:

Access the prerequisite course by prerequisiteId from coursesHashTable

Output " - ", followed by prerequisiteCourse.title, " (", and prerequisiteCourse.courseNumber, ")"

Else:

Output " - There are no prerequisites."

Output "------"

// Tree pseudocode

Function printCourseInformation(coursesTree, courseNumber):

courseNode = findNode(coursesTree, courseNumber)

If courseNode is not null:

Print "Course Number: ", courseNode.courseNumber

Print "Title: ", courseNode.title

Print "Prerequisites: "

If courseNode has children (prerequisites):

For each childNode in courseNode's children:

Print childNode.courseNumber, childNode.title

Else:

Print "No prerequisites."

Else:

Print "Course ", courseNumber, " not found."

Function numPrerequisiteCourses(coursesTree, courseNumber):

courseNode = findNode(coursesTree, courseNumber)

If courseNode is not null:

prerequisitesCount = countChildrenRecursive(courseNode)

Print "Total prerequisites for ", courseNumber, ": ", prerequisitesCount

Else:

Print "Course ", courseNumber, " not found."

**findNode** would be a function that traverses the tree to find a node with the given

**courseNumber**. Direct tree structure, which might not fully apply for complex prerequisite relationships that are better represented by a graph.

Function countChildrenRecursive(node):

If node has no children:

Return 0

Else:

count = 0

For each childNode in node's children:

count += 1 + countChildrenRecursive(childNode)

Return count Function printSampleSchedule(coursesTree):

For each courseNode in coursesTree:

InOrderTraversal(courseNode)

Function InOrderTraversal(node):

If node is not null:

If node has left child:

InOrderTraversal(node.left)

Print "Course: ", node.title, " (", node.courseNumber, ")"

If node has prerequisites (right child in a binary tree context):

Print "Prerequisites: "

InOrderTraversal(node.right)

Else:

Print "------"

## *Runtime Analysis*

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **for all courses** | 1 | n | n |
| **if the course is the same as courseNumber** | 1 | n | n |
| **print out the course information** | 1 | 1 | 1 |
| **for each prerequisite of the course** | 1 | n | n |
| **print the prerequisite course information** | 1 | n | n |
| **Total Cost** | | | 4n + 1 |
| **Runtime** | | | O(n) |

**Citations**

**https://learn.zybooks.com/zybook/CS-300-R4804-OL-TRAD-UG.24EW4/chapter/4/section/1**

**https://learn.zybooks.com/zybook/CS-300-R4804-OL-TRAD-UG.24EW4/chapter/4/section/2**

**https://learn.zybooks.com/zybook/CS-300-R4804-OL-TRAD-UG.24EW4/chapter/5/section/1**

**https://learn.zybooks.com/zybook/CS-300-R4804-OL-TRAD-UG.24EW4/chapter/1/section/10**

**https://learn.zybooks.com/zybook/CS-300-R4804-OL-TRAD-UG.24EW4/chapter/1/section/9**