Austin Frey

Professor Ling

CS-300

November 20, 2022

Milestone Four Hash Table Pseudocode

1. **File opening, reading, parsing, and formatting**

Function Parser

Pass in: file name (file), the data type (type), and the separator character (sep)

Initialize line variable for reading in data and Vector list

IF type is an eFILE

Create input file stream for file

IF file is open

WHILE input file stream does not have errors

Read line

IF line is not empty

Push line to list

END IF

END WHILE

Close file

IF file is empty

Throw error

END IF

Call: parseHeader() & parseContent()

ELSE Throw error

END IF ELSE

ELSE

Create input stream for user-entered data

WHILE there are lines to process

IF line is not empty

Push line to list

END IF

END WHILE

IF input size is 0

Throw error

END IF

Call: parseContent()

END IF ELSE

Pass out: Parser object

End function

Function parseHeader // Potentially redundant function

Pass in: None

Create stream for each line to be parsed

Create item variable for storing header names

Create course object called header to store header lines

WHILE Not at the end of the line

Add item to header course

END WHILE

Pass out: None

End function

Function parseContent

Pass in: None

Initialize vector iterator index to beginning of the file + 1 to skip the header

FOR iterator doesn’t equal file end, increment iterator

Initialize courseID,

FOR length of iterator’s line

IF less than two elements in line

Throw error

ELSE

Add each element to array for creating the course object

END IF ELSE

END FOR

Create course object and add it to courses

END FOR

Pass out: None

End function

Function validateCourseInfo

Pass in: Array of courses, prerequisite course ID being tested

FOR each element in array

IF a course ID matches prerequisite

Return true

END IF

END FOR

Return false

Pass out: Boolean representing whether or not the course ID was found

End function

1. **Course objects and data structures**

Struct Course

Pass in: Course ID, course title, and prerequisites

Initialize class variables (ID, title, prerequisites array, key,)

Pass out: Course object

End struct

Struct Node

Pass in: None OR course OR course and key

Initialize node variables (course, key, next)

Create vector of nodes

Set tableSize to a constant number

Declare hash function

Pass out: Node object

End struct

Function hash

Pass in: key

RETURN key modulo tableSize

Pass out: hash value

End function

Function addCourse

Pass in: course

Create key from course’s courseID using hash function

Create new course object

Call: insertCourse(course)

Pass out: None

End function

Function insertCourse

Pass in: course

Create hash key using hash(courseID)

Retrieve node at respective hash key index

IF node is null

Create new course using courseID and hash key

ELSE IF node’s key is the preset value

Assign course to node

SET node’s next pointer to null

ELSE

WHILE next node is not null

SET node to node’s next node

END WHILE

SET node’s next node to new node using course and hash key

END IF ELSE

Pass out: None

End function

Function printAll

Pass in: None

FOR every node in hash table

IF node’s key is not default value

Print information

CREATE nextNode from node’s next pointer

WHILE nextNode is not null

Print information

SET nextNode to nextNode’s next pointer

END WHILE

END IF

END FOR

Pass out: None

End function