Suraj Hamal

Southern New Hampshire University

Date: 06/04/2025

**Project One Milestone Three: Psedocode For Binary Tree Data Structure**

1. **Psedocode for Opening, Reading, and Validating Course File:**

FUNCTION LoadCourses(fileName, BinarySearchTree<course> courses)

OPEN file fileName FOR READING

IF file CANNOT BE OPENED THEN

PRINT "ERROR: Unable to open file"

RETURN

ENDIF

CREATE set validCourseNumbers

WHILE NOT end of file

READ line FROM file

SPLIT line INTO tokens BY ","

IF tokens.length < 2 THEN

PRINT "ERROR: Line must have at least course number and name"

CLOSE file

RETURN

ENDIF

SET courseNumber TO tokens[0]

ADD courseNumber TO validCourseNumbers

ENDWHILE

RESET file TO BEGINNING

CLEAR courses

WHILE NOT end of file

READ line FROM file

SPLIT line INTO tokens BY ","

CREATE course

SET course.courseNumber TO tokens[0]

SET course.name TO tokens[1]

SET course.prerequisites TO empty list

FOR i FROM 2 TO tokens.length - 1

IF tokens[i] NOT IN validCourseNumbers THEN

PRINT "ERROR: Prerequisite " + tokens[i] + " does not exist as a course"

CLOSE file

RETURN

ENDIF

APPEND tokens[i] TO course.prerequisites

ENDFOR

SET courses [course.courseNumber] TO course

ENDWHILE

CLOSE file

RETURN true

END FUNCTION

1. **Pseudocode for Creating and Storing Course Objects:**

STRUCTURE Course

STRING courseNumber

STRING courseName

LIST OF STRING prerequisites

END STRUCTURE

STRUCTURE Node

Course data

Node left

Node right

END STRUCTURE

CLASS BinarySearchTree

Node root

FUNCTION Insert(course)

IF root IS NULL THEN

CREATE newNode WITH course

SET root TO newNode

ELSE

CALL InsertRecursively(root, course)

ENDIF

END FUNCTION

FUNCTION InsertRecursively(currentNode, course)

IF course.courseNumber < currentNode.data.courseNumber THEN

IF currentNode.left IS NULL THEN

CREATE newNode WITH course

SET currentNode.left TO newNode

ELSE

CALL InsertRecursively(currentNode.left, course)

ENDIF

ELSE

IF currentNode.right IS NULL THEN

CREATE newNode WITH course

SET currentNode.right TO newNode

ELSE

CALL InsertRecursively(currentNode.right, course)

ENDIF

ENDIF

END FUNCTION

FUNCTION storeCourses(filename, BinarySearchTree<course> courses)

OPEN file filename FOR READING

IF file CANNOT BE OPENED THEN

PRINT "ERROR: Unable to open file"

RETURN

ENDIF

CLEAR courses

WHILE NOT end of file

READ line FROM file

SPLIT line INTO tokens BY “,”

CREATE course

SET course.courseNumber TO tokens [0]

SET course.name TO tokens [1]

SET course.prerequisites TO empty list

FOR i FROM 2 TO tokens.length – 1

APPEND tokens[i] TO course.prerequisties

ENDFOR

SET courses[course.courseNumber] TO course

ENDWHILE

CLOSE file

RETURN true

END FUNCTION

1. **Pseudocode for Searching and Printing Course Information:**

FUNCTION SearchCourse(BinarySearchTree<Course> courses, String courseNumber)

SET course TO SEARCH courseTree FOR courseNumber

IF courses IS NOT null THEN

PRINT course.courseNumber + “,” + course.courseName

IF course.prerequisites IS NOT empty THEN

SET prerequisiteString TO “Prerequisites: ”

SET isFirst TO true

FOR each prerequisite IN course.prerequisites

IF isFirst THEN

SET prerequisiteString TO prerequisiteString + prerequisite

SET isFirst TO false

ELSE

ADD prerequisiteString TO prerequisiteString + “, ” + prerequisite

ENDIF

ENDFOR

PRINT prerequisiteString

ELSE

PRINT “Prerequisites: None”

ENDIF

ELSE

PRINT ERROR IF COURSE NOT FOUND

ENDIF

END FUNCTION