Suraj Hamal

Southern New Hampshire University

Date: 06/11/2025

**Project One:Psedocodes for Menu and to Print Courses**

1. **Psedocode For a Menu for All Three Data Structures (Rubric 2):**

FUNCTION DisplayMenu()

PRINT “Welcome to the Course Planner”

PRINT “1. Load Courses”

PRINT “2. Print Course List”

PRINT “3. Find and Print Course”

PRINT “4. Exit the Program”

END FUNCTION

FUNCTION Main(arguments)

DECLARE STRING csvPath

DECLARE STRING courseKey

DECLARE INTEGER choice = 0

DECLARE CLOCK ticks

DECLARE Bid bid

IF number of arguments IS 2 THEN

SET csvPath TO argument[1]

ELSE IF number of arguments IS >= 3 THEN

SET csvPath TO argument[1]

SET courseKey TO argument[2]

ELSE

SET csvPath TO "courses.csv"

ENDIF

PRINT "Enter course key "

READ courseKey FROM USER

// Choose data structure here

DECLARE courses AS (Vector or HashashTable or BinarySearchTree) OF Course

SET choice TO 0

WHILE choice IS NOT 9 DO

CALL DISPLAY Menu()

PROMPT user to "Enter choice:"

READ choice

SWITCH (choice)

CASE 1:

PRINT “Loading courses…”

SET ticks TO CURRENT CLOCK TIME

CALL loadCourses(csvPath, courseTree)

SET ticks TO CURRENT CLOCK TIME - ticks

DISPLAY ticks IN clock ticks AND seconds

BREAK

CASE 2:

CALL printCourseList(courses)

BREAK

CASE 3:

SET ticks TO CURRENT TIME

SET course TO courseTree.Search(courseKey)

SET ticks TO CURRENT TIME - ticks

PROMPT "Enter course number"

READ courseKey

// For Vector and Binary tree

CALL searchCourse(courses, courseKey)

// For BINARY TREE

SET course TO courseTree.Search(courseKey)

IF course IS NOT NULL THEN

CALL displayCourse(course)

ELSE

DISPLAY "Course not found"

ENDIF

BREAK

CASE 9:

DISPLAY "Goodbye."

BREAK

DEFAULT:

DISPLAY "Invalid Option. Try Again With Options 1, 2, 3, or 9."

ENDSWITCH

ENDWHILE

END FUNCTION

1. **Psedocode to print out the list of the courses in alphanumeric order for all the three data structures.**

**Vector Data Structure:**

FUNCTION PrintAllCourses\_Vector(Vector<Course> courseVector)

SORT courseVector BY courseNumber in ascending order

CALL sort(courseVector, compareByCourseNumber)

FOR EACH course IN courseVector

PRINT course.courseNumber + ": " + course.courseName

IF course.prerequisites IS NOT EMPTY THEN

PRINT " Prerequisites: "

FOR EACH prereq IN course.prerequisites

PRINT " " + prereq

ENDFOR

ENDIF

ENDFOR

END FUNCTION

**HashTable Data Structure:**

FUNCTION PrintAllCourses\_HashTable(HashTable<Course> courseTable)

DECLARE List<Course> courseList

FOR EACH bucket IN courseTable

FOR EACH course IN bucket

APPEND course TO courseList

ENDFOR

ENDFOR

SORT courseList BY courseNumber in ascending order

CALL sort(courseList, compareByCourseNumber)

FOR EACH course IN courseList

PRINT course.courseNumber + ": " + course.courseName

IF course.prerequisites IS NOT EMPTY THEN

PRINT " Prerequisites: "

FOR EACH prereq IN course.prerequisites

PRINT " " + prereq

ENDFOR

ENDIF

ENDFOR

END FUNCTION

**Binay Search Tree Data Structure:**

FUNCTION PrintAllCourses\_BST(Tree<Course> courseTree)

CALL InOrder(courseTree.root)

END FUNCTION

FUNCTION InOrderTraversal(Node node)

IF node IS NOT NULL THEN

CALL InOrderTraversal(node.left)

PRINT node.course.courseNumber + ": " + node.data.courseName

IF node.data.prerequisites IS NOT EMPTY THEN

PRINT " Prerequisites: "

FOR EACH prereq IN node.data.prerequisites

PRINT " " + prereq

ENDFOR

ENDIF

CALL InOrderTraversal(node.right)

ENDIF

END FUNCTION