Brandon S. Luellen

CS – 300 DSA: Analysis and Design

Nathan Lebel

October 8, 2023

1. Pseudocode for reading and checking file format errors:

FUNCTION readFile(File f, lines[])

courseNumbers[], courseTitles[], prerequisites[], line

i = 0, j = 0

Flag = TRUE

WHILE (NOT END OF FILE f)

courseInfo[] = SPLIT(READLINE(f, line), DELIMITER = ",")

APPEND line TO lines

IF (LENGTH of courseInfo < 2)

Flag = FALSE

BREAK

END IF

courseNumbers[i] = courseInfo[0]

courseTitles[i] = courseInfo[1]

INCREMENT i

IF (LENGTH of courseInfo > 2)

FOR k = 2 to LENGTH of courseInfo

prerequisites[j] = courseInfo[k]

INCREMENT j

END FOR

END IF

END WHILE

IF Flag == TRUE

FOR each P in prerequisites

IF P NOT IN courseNumbers

Flag = FALSE

BREAK

END IF

END FOR

END IF

RETURN Flag

END FUNCTION

2. Pseudocode for creating course objects and storing them:

CLASS Course

Number: String

Title: String

Prerequisites[]: String[]

CONSTRUCTOR Course(line)

Number = SPLIT(line, DELIMITER = ",")[0]

Title = SPLIT(line, DELIMITER = ",")[1]

IF LENGTH of SPLIT(line, DELIMITER = ",") > 2

Prerequisites = SPLIT(line)[2 to LENGTH of SPLIT(line, DELIMITER = ",")]

END IF

END CONSTRUCTOR

END CLASS

FUNCTION createObject(Courses <Course>, File f)

Lines[] = " "

IF readFile(f, Lines) == TRUE

FOR each Line in Lines

APPEND NEW Course(Line) TO Courses

END FOR

ELSE

PRINT("File cannot be read")

END IF

END FUNCTION

3. Pseudocode for searching and printing course information:

FUNCTION MAIN()

Filename = INPUT()

File F = NEW File(Filename)

Courses <Course>: vector

CALL : createObject(Courses, F)

CourseNumber = INPUT()

FOR all Courses

IF the course is the same as CourseNumber

PRINT out the course's information

IF course has prerequisites

FOR each prerequisite P

PRINT P as prerequisite course information

END FOR

END IF

END IF

END FOR

END FUNCTION

4. Evaluation of runtime and memory:

Runtime complexity: O(n^2) due to the nested loops and potential

linear search within prerequisites.

Memory usage: The memory consumption primarily depends on the size

of the input file and the number of courses, but it does not have

any significant issues with memory usage for moderate-sized datasets.

5. Analysis of data structures:

Vectors: Suitable for storing a list of courses, but may not be efficient

for searching and maintaining alphabetical order.

Hash tables: Efficient for quick lookups but not suitable for maintaining

alphabetical order or handling duplicate keys.

Trees: Ideal for maintaining alphabetical order and handling hierarchical

relationships, making them a good choice for this application.

6. Recommendation:

Based on the analysis, a tree structure (e.g., a binary search tree) is

recommended for storing and managing the list of courses. It provides

efficient alphabetical order traversal and allows for easy retrieval of

course information and prerequisites, making it well-suited for the given

requirements.