1. Define a class named `Course` with instance variables: `courseNumber`, `name`, and `prerequisites`.

2. Define a function called `loadData` that accepts a file path as a parameter.

3. Open the file specified by the file path and store the file handle in a variable.

4. Create an empty vector called `courses` to store `Course` objects.

5. Read the first line of the file and store it in a variable.

6. Loop until the end of the file is reached:

a. Split the line into an array using a comma as the delimiter.

b. If the array has fewer than 2 elements, raise an error and terminate the program (each line must have at least two fields).

c. Create a new `Course` object and set its `courseNumber`, `name`, and `prerequisites` variables based on the array's contents.

d. Add the `Course` object to the `courses` vector.

e. Read the next line of the file and store it in the same variable.

7. Close the file.

8. Return the `courses` vector.

1. Define a function named `printCourses` that takes a tree structure as a parameter.

2. Traverse the tree using a depth-first traversal approach.

3. For each node encountered:

- Print the `courseNumber` and `name`.

- If the node has prerequisites, print the `courseNumber` of each prerequisite.

4. Continue the traversal and repeat the above steps for all nodes in the tree structure.

// Open file "courseInformation.csv" for reading

open file "courseInformation.csv" for reading

IF file does not exist

Display ERROR and exit

ELSE

create empty tree data structure "courses"

WHILE a line exists in file

read line

split line into token

IF number of tokens is < 2

display ERROR

continue to next line

END IF

IF first token not a number

display ERROR

continue to next line

END IF

IF a third token exists

IF third token not a number

Display ERROR

continue to next line

END IF

check if third token exists as a node in "courses"

IF not

Display ERROR

continue to next line

END IF

END IF

create new course object with first token as course number, second token as course title, and third as prerequisite

insert course object as a node in "courses" with the prerequisite as its parent node

END WHILE

END IF

// Close the file

CLOSE FILE

// Define Course class

class Course

courseNumber

courseTitle

prerequisite

constructor(courseNumber, courseTitle, prerequisite)

this.courseNumber = courseNumber

this.courseTitle = courseTitle

this.prerequisite = prerequisite

END constructor

END class

// Define function to display course information

function displayCourses(node)

display courseNumber, courseTitle, and prerequisite of current node

IF node has children

for each child of node

call displayCourses function with child as argument

END FOR

END IF

END function

// Call displayCourses function with root node of "courses" tree

call displayCourses function with root node of "courses" tree data structure as argument