Edwin Jones

CS300 3-3: Project One Milestone One

//Defaults to public members

struct CourseInfo {

string code

string name

//accounts for 0-N prerequisites

vector<string> prerequisites

void printSortedVector(vector<CourseInfo> &courses) {

sort using built in sort function

for each course in courses

print course code

print couse name

for each prereq in course prerequisites

if prereq length is 0

continue

if prereq is the last in the vector

print prereq

else

print prereq with a following comma

}

//Read CSV file then add row to vector of struct courseInfo

void loadCSVToVector(string csvFile, vector<CourseInfo> &courses) {

ifstream file(csvFile)

if file is not open

throw an error

string line

while getline(file, line)

if the line is empty

continue

if the line doesn't end with a comma

add a comma to the end

vector<string> tokens

stringstream ss(line)

string token

while getline(ss, token, ',')

add token to end of tokens vector

if the tokens vector size is < 2

throw an error

create CourseInfo course instance

set course.code = token at index 0 of tokens vector

set course.name = token at index 1 of tokens vector

for i=2, i++ while i is < tokens vector size

if tokens vector at i is not empty

add token at index i to end of course.prerequisites

add course to end of courses vector

close file

}

// Verify that each unique course prerequisite is listed as a course

void validateCourses(vector<CourseInfo> &courses){

vector<string> tempPrereqs

vector<string> tempCourses

//courses listed to check prerequisite courses are listed

for each CourseInfo course instance in the courses instance

add course.code to the end of the tempCourses vector

//check each course for their prerequisites

for each CourseInfo course instance in the courses instance

//check if prerequisite course is listed as course

for each string prereq in course.prerequisites

if prereq is in tempCourses

continue

else

throw an error

}

// Search for course given course code as search parameter

void searchCourse(vector<CourseInfo> people, string searchItem)

for each CourseInfo course instance in courses

if each course.code = searchItem

print course.code

print course.name

for each string prereq in course.prerequisites

if prereq length is 0

continue

print prereq

return

print “Course code not found”

main {

set string csvFile = csv file location

create vector<CourseInfo> courses

set userChoice as 0

while userChoice != 9

print menu

set userChoice to user input

switch(userChoice)

case 1: // Load the file data into the data structure

call the loadCSVToVector(csvFile, courses) method

case 2:

call the validateCourses(courses) method

case 3: // Print an alphanumerically ordered list

call the printSortedVector(courses) method

case 4: // Print the course title and the prerequisites for any individual course

call the searchCourse(people, get user search parameter) method

return 0

}