# CS 300 Pseudocode Document

## Pseudocode is used to open files, read data from the file, parse, and check for file format errors.

The application will first try to open a CSV file and will parse it line by line if the file exists and is readable. It will validate that each line is greater or equal to two items on a successful line reading. Otherwise, it will throw an error, and (Quit/Move to the next line??).  
For every 3rd, 4th, …Nth element in each parsed row (end of line), it will validate each element that such exists in Vector<Course> courses->cource->courceName. If an element was found in courses, create a new Course object, populate it with the parsed line, and store it in the courses Vector. Otherwise, (Quit/Move to the next element??).

Main():

filePath = "path/to/course\_data.csv"

Try to open file:

Parser Line = csv::Parser(filePath);

For each Line in the filePath, Try reading a line:

If line length ≥ 2:

Construct a New class Course object:

courseNumber = lineData[0]

courseName = lineData[1]

For every Item in Line ≥ lineData[2]

Validate prerequisites:

If lineData[2, 3, N] in Vector<Course> courses->cource>courceName:

Course object prerequisite[0,1,…,N] = lineData[2 to end]

Append New class Course object to Vector<Course> courses.

Catch Can’tReadLineError: print("Error: Line not read at…")

Catch Can’tOpenFileError: print("Error: File not found at…")

## Pseudocode to show how to create course objects and store them in the appropriate data structure.

* The data will be stored in a vector of Course objects:

Vector<Course> courses = []

* Each object per one read line from the file will be structured and initialized as follows:

class Course object:

string courseNumber

string courceName

list<string> prerequisites

Course constructor(cNum, cNam, cpReq[]):

this courseNumber = cNum;

this courceName = cNam;

this prerequisites = cpReq[];

## Pseudocode that will search the data structure for a specific course and print out course information and prerequisites.

The function searchCourse(Vector<Course> courses, String course number)will iterate through each Cource item in the Vector and compare it to the courseNumber parameter. If a match is found, it will print to screen courseNumber, courceName attributes, and all or none of the prerequisites from the list if such exist.

Function SearchCourse(vector<Course> courses, string CNumber):

For a course in courses:  
if course->courseNumber == CNumber:

print(course->courseNumber, course->courceName)

if course->prerequisites is not null:

for prerequisite in prerequisites:

print(prerequisite[0], prerequisite[Nth])

Return;