# CS 300 Vector Pseudocode Document

## Function Signatures

Below are the function signatures that you can fill in to address each of the three program requirements using each of the data structures. The pseudocode for printing course information, if a vector is the data structure, is also given to you below (depicted in bold).

Declare a vector of Course called course;

Int main(){

Start

While True:

Print Menu header;

Print choice 1. Load File

Print choice 2: Display All

Print choice 3: Print Course

Print choice 4: Exit

Declare an integer variable called choice

Ask for user input;

Switch user input:

Case 1:

Decalre a string variable called filePath;

Prompt user for filepath

Call the loadAndReadfile function passing filePath in the parameter;

Display message letting user know file was successfully loaded;

Break;

Case 2:

Call the displayAll function;

Break;

Case 3:

Declare a string variable and name it courseNumber;

Prompt user to enter a course number

Call the display function passing courseNumber in the parameter

Break;

Case 4:

Print message telling user the system is exiting;

Return 0;

Break

Default:

Print invalid choice;

End Switch

End While

End

}

Vector<string> loadAndReadFile(Vector<Course> courses, Course c) {

Start

Create an fstream object

Open the file

If the file is not opened:

Display an error message

Exit

End If

Else

For each line of course in courses:

Read a line from the file

Create a token to parse course number, title and prereqs

If the tokens of each line less than two

Display an error message and continue to read file

Until EOF has been reached.

End If

For int i is Equal to 2:

Check if there is a prerequisite at position 2

Display “ No prereq found for course”

Increment i until EOF is reached

End For

End Else

End

}

Course createCourse(string file){

Start

Create a new course object assuming the struct has been

Initialized.

For int I is equal to zero and I is less than the rows in the

File:

Add course number to the file in row 0

Add the course name to row 1

Add the course prereq to row 2

Append each course

End For

End

}

Course Search(string courseNum){

Declare vector<string> course;

While course.courseNum is not equal to NULL:

If course at position 0 is equal to courseNum

Then display course information

For each prereq in course:

Display prereq information

End For

End If

End While

End

}

Void displayAll(){

If courses is empty:

Print Error message;

End if

Else:

Call the sort function and pass courses.begin, courses.end and array of two Course objects;

Return firstObject.courseNumber less than secondObject.courseNumber;

}

End else;

For each Course in courses:

Print course number and title;

End for Each

End

## Example Runtime Analysis

When you are ready to begin analyzing the runtime for the data structures that you have created pseudocode for, use the chart below to support your work. This example is for printing course information when using the vector data structure. As a reminder, this is the same pairing that was bolded in the pseudocode from the first part of this document.

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **Create an fstream object** | 1 | 1 | 1 |
| **Open the file** | 0 | 0 | 0 |
| **If the file is not opened** | 1 | 1 | 1 |
| **Display an error message** | 1 | 1 | 1 |
| **For each line of course in courses** | 1 | n | n |
| **Read a line from the file** | 1 | n | n |
| **If the tokens of each line less than two** | 1 | n | n |
| **Display an error message and continue to read file** | 1 | n | n |
| **Until EOF has been reached** | 0 | 0 | 0 |
| **For int i is Equal to 2** | 1 | n | n |
| **Check if there is a prerequisite at position 2** | 1 | n | n |
| **Display “ No prereq found for course”** | 1 | n | n |
| **Increment i until EOF is reached** | 1 | n | N |
| **Total Cost** | 8n+3 | | |
| **RunTime** | O(n) | | |