# CS 300 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).

// Course Structure pseudocode:

Structure Course{

Course number

Course description

List of prerequisites

}

//File IO pseudocode:

Void CSVParser {

Get and open CSV file with data in it

While you have not reached the end of the file

Get next line of data

Parse data by splitting at the comma and construct a data object course out of the data.

Add this course object to vector

Move on to the next line of data and continue

Close the file.

}

// Vector pseudocode

int numPrerequisiteCourses(Vector<Course> courses, Course c) {

totalPrerequisites = prerequisites of course c

for each prerequisite p in totalPrerequisites

add prerequisites of p to totalPrerequisites

return number of totalPrerequisites

}

void printSampleSchedule(Vector<Course> courses) {

for each course in courses

if all prerequisites have been met by student or numPrerequisiteCourses is 0

print out this course to screen.

}

void printCourseInformation(Vector<Course> courses, String courseNumber) {

**for all courses**

**if the course is the same as courseNumber**

**print out the course information**

**for each prerequisite of the course**

**print the prerequisite course information**

}

// Hashtable pseudocode

int numPrerequisiteCourses(Hashtable<Course> courses, string courseNumber) {

get the key from the course number

run through hash function

return number prerequisites

}

void printSampleSchedule(Hashtable<Course> courses) {

First get number prerequisites this will be come key.

Run this key through hash function.

Go to bucket at this index and print out the courses located

In this bucket.

}

void printCourseInformation(Hashtable<Course> courses, String courseNumber) {

Take course number string and turn into a key

Run key through hash function to get the index

Go to this index

If the key of the item matches the key you are looking for:

Print this course information.

Return.

Else:

Set up current course variable = first course at index

While the current course isn’t null:

If the current course’s key = keyed course Number

Print this course information

Return

}

// Tree pseudocode

int numPrerequisiteCourses(Tree<Course> courses) {

Get the course number

Start at current = root.

If current’s course number matches the course number

Return number prerequisites for this course

Else If the course’s number is greater than at the current node

then move to the right and compare here.

Else

Move to left and compare here

Once found:

Return number prerequisites

}

void printSampleSchedule(Tree<Course> courses) {

Start at root.

If course has no prerequisites, print to screen.

Continue traversing down left side of tree, printing the courses that have no prerequisites out.

Then, down the right side, if the course prerequisite has been met, print out the course information.

}

void printCourseInformation(Tree<Course> courses, String courseNumber) {

Make variable called current and set equal to root of tree.

If current course is empty -> course was not found.

If current’s course number matches courseNumber:

Print this course.

Else:

If current course’s number is larger than courseNumber,

Move to the current’s left and repeat comparison with this course.

Else:

Move to current’s right and repeat comparison with this course

(this method will recursively call to itself and repeat these steps until it finds the course or gets to the bottom of the tree and does not find it)

}

## 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** |
| --- | --- | --- | --- |
| **for all courses** | 1 | n | n |
| **if the course is the same as courseNumber** | 1 | n | n |
| **print out the course information** | 1 | 1 | 1 |
| **for each prerequisite of the course** | 1 | n | n |
| **print the prerequisite course information** | 1 | n | n |
| **Total Cost** | | | 4n + 1 |
| **Runtime** | | | O(n) |