# 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).

// 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

print number of totalPrerequisites

}

void printSampleSchedule(Vector<Course> courses) {

if courses.containers(courseNumber):

course = courses.get(CourseNumber):

print out the course information

for each prerequisite of the course:

prerequisiteCourse = courses.get(prerequisite)

print the prerequisite course information

}

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) {

course = findCourse(courses, courseNumber)

if course is null:

return 0

totalPrerequisites = new HashSet<String>()

queue = new Queue<String>()

queue.enqueue(course.prerequisites)

while queue is not empty:

prerequisite = queue.dequeue()

if prerequisite not in totalPrerequisites:

totalPrerequisites.add(prerequisite)

prerequisiteCourse = findCourse(courses, prerequisite)

if prerequisiteCourses is not null:

queue.enqueue(prerequisiteCourse.prerequisites)

return totalPrerequisites.size()

}

// Print course and prerequisites for a sample schedule

void printSampleSchedule(Hashtable<Course> courses) {

for each courseNumber in courses.keys:

printCourseInformation(courses, courseNumber)

}

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

course = findCourse(courses(courses.root, courseNumber)

if course is not null:

print out the course information

for each prerequisite of course:

prerequisiteCourse = findCourse(courses.root,prerequisite)

print the prerequisite course information

}

// Tree pseudocode

// Count total prerequisites for a course

int numPrerequisiteCourses(Tree<Course> courses) {

course = findCourse(courses, courseNumber)

if course is null:

return 0

totalPrerequisites = new HashSet<String>()

queue = new Queue<String>()

queue/enqueue(course.prerequisites)

while queue is not empty:

prerequisite = queue.dequeue(

if prerequisite is not in totalPrerequisites:

totalPrerequisites.add(prerequisite)

prerequisiteCourse = findCourse(course, prerequisite)

if prerequisiteCourse is not null:

queue.enqueue(prerequisiteCourse.prerequisites)

return totalPrerequisites.size

}

// Print course and prerequisites for a sample schedule

void printSampleSchedule(Tree<Course> courses) {

for each course in courses:

printCourseInformation(courses, course.courseNumber)

}

// Print a course and its prerequisites

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

course = findCourse(courses, courseNumber)

if course is not null:

print out the course information

for each prerequisite of course:

prerequisiteCourse = findCourse(courses, prerequisite)

print the prerequisite course information

}

## 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) |