Frank Colabella

Project 1

CS 300

**Pseudocode**

**Vector Pseudocode**

Open file "course information.txt"

Read and Parse "course information.txt"

if file not found

Print "error"

Exit

End if

Ensure at least two parameters per line

Ensure prerequisites have another line with that course number

Define Structure Courses

courseID

courseName

prerequisite

Initialize vector CourseList

While Getline from file

read line by line

Split line into tokens, using comma as delimiter

Add the elements to the structure

Push Back into vector

Get input to search for course

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

**Hash Table Pseudocode**

Open file "course information.txt"

Read and Parse "course information.txt"

if file not found

Print "error"

Exit

End if

Ensure at least two parameters per line

Ensure prerequisites have another line with that course number

Define Structure Courses

courseID

courseName

prerequisite

Initialize vector CourseList

Initialize hash function to set key values to vector

While Getline from file

read line by line

Split line into tokens, using comma as delimiter

Add the elements to the structure using hash function

Get input to search for course

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

**Binary Search Tree Pseudocode**

Open file "course information.txt"

Read and Parse "course information.txt"

if file not found

Print "error"

Exit

End if

Ensure at least two parameters per line

Ensure prerequisites have another line with that course number

Define Structure Courses

courseID

courseName

prerequisite

Initialize binary search tree BST

While Getline from file

read line by line

Split line into tokens, using comma as delimiter

Add the elements to the structure

Use recursive function to add nodes to BST

Get input to search for course

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

**Menu Pseudocode**

WHILE input does not equal 9

SWITCH input

CASE 1

Load data into data structure

BREAK

CASE 2

Print courses alphanumerically

BREAK

CASE 3

Print course title and prerequisites for individual course

BREAK

**Sorting Pseudocode**

SORT vector with sort function, using default comparison parameter

FOR course list

OUTPUT courses

**Runtime Analysis and Recommendation**

The runtime for creating the vector and hash table data structures is O(n), and the runtime for the binary tree is O(log n). This aligns with the nature of the data structures. Searching through a vector can be significantly slow because each node is checked in order. Hash tables have the potential to be fast but can have slow worst case scenarios due to the arbitrary nature of hashing. Binary search trees have faster worst case scenarios due to the use of recursion and comparisons that take place when traversing down a tree, eliminating many unnecessary checks. Therefore, my recommendation for storing the required data is a binary search tree.