James Peace

CS 300 Analysis and Design

Project one submission

**Vector Data:**

Call file to be opened

If file exists

Open file

Read column information

If courseNumber exists

Store variables in nodes (courseNumber, courseName, prerequisiteN)

Assign courseNumber to head of each vector

Print course information

Find head node (courseNumber)

Print courseNumber

Print courseName

If prerequisite exists

Print PrerequisiteN information

Loop until entire file is printed

**Hash Tables:**

Call file to be opened

If file exists

Open file

Read column information

If courseNumber exists

Store variables in table (courseNumber, courseName, prerequisiteN)

Assign courseNumber to key

Print course information

Find key node (courseNumber)

Print courseNumber

Print courseName

If prerequisite exists

Print PrerequisiteN information

Loop until entire file is printed

**Tree Data Structure:**

Call file to be opened

If file exists

Open file

Create root tree (courses)

Create tree subroot nodes (courseNumber, name, prerequisite)

Read Nodes and create tree for each Course

Validate prerequisite exists

If prerequisite listed (assign temp node)

Search from beginning of tree

Once match courseNumber found

Insert temp node right subtree of matching courseNumber

If prerequisite not listed

Move to next course

Print course information

Find tree root node Courses

Search current course node

Print courseNumber

Print courseName

If prerequisite exists

Print PrerequisiteN information

Loop until entire tree information is printed.

**Menu:**

Start program

Print menu while input not equal to 4 (Use a switch/case)

Print “1. Load Data Structure.”

“2. Print Course List.”

“3. Print Course.”

“4. Exit”

“What would you like to do?”

If option 1

Call load data function

Read file

If option 2

Call Print course list function

If option 3

Call search course function

If no course exists

Print “Course does not exist”

If course exists

Print courseNumber, courseName, prerequisiteN

If option 4

End program

**Courses in numerical order:**

If current node is greater than previous node

Move to left of current node

If current node is less than previous node

Move to right of current node

Loop until all nodes are in order

Print list.

I’m unsure how to definitively determine the runtime analysis for each of these type lists. However, if I understand what the question is, it seems to me that given the example that each line of code counts as one; then given 8 courses each loop would have to run at least 8 times. If I give print each course a value of 1 then it appears that Hash Tables or vector lists are the most memory efficient.

However, in previous assignments the fastest runtime in application was from Linked Lists. Hash tables were very slow, and Binary tree was almost as fast as Linked Lists. Linked lists loading the CSV file in just .012 seconds while Binary tree loading the same file in .045 seconds; Hash tables was over 2.45 seconds load time. I’m suddenly getting a memory error when trying to run my Vector Sorting program from Module 2, so I cannot compare these.

So, I’m unclear as to how to use the runtime analysis to determine which would be the most efficient use of resources. Since I’m getting an error when trying to run Vector Sorting, and it doesn’t appear that we are allowed to use the Linked List option for this project; I will write the code using Binary trees.