Daniel Anderson

System Analysis and Design

02/09/2023

**Project One**

When looking at which data structure works best for course information there is a lot of different things that you need to look at to ensure you get the best results out of your data structures. After reviewing the three data structures types I have determined that a binary tree would work best because of there are numbers attached to each class that would make it easier to search and sort by. Another great advantage of the Binary tree is the speed at which class will be searched because they will either be to the left or right of the beginning node which already reduces the search time based on the class needed. There is however a downside to the binary tree and that is when searching for the prerequisites for each class since the main sorting system would be on class number. Everything else the binary tree will be baseline faster and better than every other system because of how it sorts the information out.

**Psuedocode**

OPEN file “filename.txt”

IF file not found THEN

OUTPUT “File not found.”

IF file found THEN

OUTPUT “File Found”

READ file

GET LINE

ADD line to NODE

FORMAT LIST

COURSE NUMBER

COURSE NAME

PREREQUISITE1

CHECK course number

PREREQUISITE2

CHECK course number

GET next LINE

IF no more lines

SAVE list

Close file

CREATE tree

IF node = null

Current->node = Course Number

IF node != null

GET next Course Number

IF Course Number < Current(node)

ADD NODE left of current(node)

IF Course Number > current(node)

ADD NODE right of current(node)

IF Course Number = null

OUTPUT “All Classes sorted”

DISPLAY node

OUTPUT “Course Number, Course Name, Prerequisite 1, Prerequisite 2”

SEARCH node

OUTPUT options

OUTPUT “1. Search by Course Number”

OUTPUT “2. Search by Course Name”

OUTPUT “3. Search by Prerequisite 1”

OUTPUT “4. Search by Prerequisite 2”

OUTPUT “5. Exit”

OUTPUT "Enter Choice:”

GET user input

IF input = 1

GET user input

IF input= node(course number)

OUTPUT course information

IF input != node(course number)

OUTPUT “Course not found”

IF input = 2

GET user input

IF input = node(course name)

OUTPUT course information

IF input != node(course name)

OUTPUT “Course not found”

IF input = 3

GET user input

IF input = node(prerequisite 1)

OUTPUT course information

IF input != node(prerequisite 1)

OUTPUT “Course not found”

IF input = 4

GET user input

IF input = node(prerequisite 2)

OUTPUT course information

IF input != node(prerequisite 2)

OUTPUT “Course not found”

IF input = 5

OUTPUT “Goodbye!”

PRINT list

OUTPUT options

OUTPUT “1. Print by Course Number”

OUTPUT “2. Print by Course Name”

OUTPUT “3. Print by Prerequisite 1”

OUTPUT “4. Print by Prerequisite 2”

OUTPUT “5. Exit”

OUTPUT "Enter Choice:”

GET user input

IF input = 1

GET user input

IF input= node(course number)

PRINT course information

IF input != node(course number)

PRINT “Course not found”

IF input = 2

GET user input

IF input = node(course name)

PRINT course information

IF input != node(course name)

PRINT “Course not found”

IF input = 3

GET user input

IF input = node(prerequisite 1)

PRINT course information

IF input != node(prerequisite 1)

PRINT “Course not found”

IF input = 4

GET user input

IF input = node(prerequisite 2)

PRINT course information

IF input != node(prerequisite 2)

PRINT “Course not found”

IF input = 5

OUTPUT “Goodbye!”

**Runtime Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Line Cost** | **# of times executed** | **Total Cost** |
| For all class | 1 | n | n |
| If course is the same as course number | 1 | 1 | 1 |
| Print out course information | 1 | 1 | 1 |
| For each prerequisite of the course | 1 | n | n |
| Print the prerequisite course information | 1 | n | n |
| **Total Cost** |  |  | 3n+2 |
| **Runtime** |  |  | O(n) |