Akhad Alimov

PSUEDO-CODE

1. Print a list of all the Computer Science courses in alphanumeric order.

ReadFile :

if file name is not found

cout << “file cant be found please try again”

go back to menu (or exit)

Parse line:

for loop readLine

if parameterCount < 2

cout << “needs at least 2 parameter per line”

exit()

while courseExists

if course does not exist

exit

check for course number, title, prerequisites

parseline

CreateCourse:

New course = course number, course title, course prerequisites

Print course info (course number, course title, course prerequisites)

1. Menu

Menu

dataStr = “Load Data Structure”

courseList = “Print Course List”

printCourse = “Print course + prerequisites”

exit = “exit”

---

loadData :

ReadFile (“file.csv”)

ParseFile (“file.csv”)

If (ParseFile) {

Vector sort

//for simplicity I will put pseudocode here but it would be in its own class

For loop:   
 swap false

If list index > list index1

Swap index

Swap true

printCourse: Print course info with given ID or name

exit = exit ()

AlphanumericSorting:

For loop:

Inner for loop:

swapNum1 = starting

swapNum2 = following

if swapNum1 > swapNum2

swap

//print can be called through the menu

Akhad Alimov

Runtime Analysis

Opening the file is very efficient as it is simple task so it would be Big O-1. Reading and parsing file is O m\*n as it uses 2 elements to read and parse. And to create course objects would be O m as it’s only 1 element. The worst runtime would be within and due to the parsing and reading the file. For alphabetic sorting, I chose to go with vector as it was simple to me and works okay, but I think binary tree would be more efficient in this case. The comparisons and sorting would be more difficult to implement, but the overall the performance of the program will improve. To modify such as adding, removing, searching I would go for hash tables since it has reliable comparison time complexity in all scenarios given.