CS 6103D Software Systems Laboratory

PROBLEM 1D 20-AUG-2019

The objective is to learn the following:

- representation of directed graph using adjacency list
- implement different topological sort algorithms
- Understand & implement Trie data structure.
- programming in Java

Submission date: on or before 26.08.2018 Monday 1:00 pm
Submission: a single .tar file named as per the format
P1D < FIRST NAME > < ROLLNO > .tar

1. The problem is to represent the course prerequisite information. A course can have one or more other courses as prerequisites. A student can register for a course only

   a. If he/she has completed all the prerequisite courses for that particular course.

   OR

   b. If he/she has completed at least 50% of all the prerequisite courses with avg. 8.5 CGPA.

   Indicate the course prerequisite information using a Directed Acyclic Graph (DAG). Each course is to be represented using a vertex in the graph. If course c_i is a prerequisite for course c_j, that information is represented by a directed edge from c_i to c_j. Represent the graph using adjacency list representation, preferably with Java HashSet and HashMap. Give a topological sort order of the DAG by

   1. implementing the algorithm given in section 22.4 of CLRS
   2. using the method given in problem 22.4-5 of CLRS

   Reference: T. H. Cormen, C. E. Lieserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*, PHI Learning, 3rd edition, 2010.

2. Find the common courses between two or many given students that they can register in above question. (Use Java Collection methods).

   Reference: The Collections Framework from The Complete Reference Java by Herbert Schildt

3. Implement printRegisteredStudents(String course, String x) method which display unique names of all registered students in the given course whose names are starting with string x (print all names which have string x as prefix of registered students) **using Trie data structure**.If x is null then print all names. (Use insert() and search() methods of Trie data structure with java collections framework).

   References:
   https://en.wikipedia.org/wiki/Trie , https://www.geeksforgeeks.org/auto-complete-feature-using-trie/, https://www.geeksforgeeks.org/trie-insert-and-search/, https://www.hackerearth.com/practice/data-structures/advanced-data- structures/trie-keyword-tree/tutorial/ , https://medium.com/basecs/trying-to-understand-tries-3ec6bede0014