Laboratory 8

COMP 271

Movie Database

We discussed the movies.txt file last class, along with some Java code we developed for representing, building, and querying an undirected graph.

You will easily understand what to do for this assignment and be able to get something going code-wise quickly, but longer execution times may have you going back to the drawing board. If you want to work with a friend in class, you are welcome to. I would further encourage mutual discussions between all of you to get this done correctly and efficiently. But the code collaboration needs to be confined to a maximum of two team members for each submission. So again, class-wide conversations are ok but not code collaboration.

Here is what needs to be done and there is flexibility in your coding choices.

1. The vertex set will consist of all people (names) who are featured in the movies (the cast) **and** the names of the movies (with the year). The edges will go between the cast members and the movies in the sense that if a person (actor) is featured in a movie then there is an edge between the two. Obviously therefore, there will not be any edge present between the actors OR between the movies.
2. Each line of the text file is to be read as a String with nextLine() and then separated out using the split("/") method. The first one in the separated out array is always the movie name and the rest are all people names. We already have developed code for this part that you can use.
3. It is recommended that as you read the movies file into memory, you will generate for each line of the file, the vertices and the edges for that line right away. Basically, you will first call addVertex() before calling addEdge() between a pair of previously created vertices.
4. It is a good idea to find and use a mechanism (some data structure or attributes) to distinguish the two types of vertices in the graph, the movies vs the cast.
5. **Do not add any code to the Graph class with regard to movie text file information. The Graph class is a general purpose utility that can used to build a graph for any type of appropriate data. This means that you will create separate Java class file(s) for doing this assignment.**
6. Feel free to use as many data structures as needed (subject to your computer's memory limits) to make the code time-efficient.
7. The first goal is to build the graph of movies and cast members featured in the movie. And in that process, build auxiliary data structures to answer questions efficiently.
8. You will then build code to answer the following questions (via suitable methods).
9. How many unique movies are there?
10. How many unique cast members are there? You can assume that two people with identical names are the same people – which we will agree as a bit of oversimplification.
11. How many edges are there?
12. Given a cast member, print the list of movies in which the cast member is featured in.

13. Write a method *public void coCast (int minMovies)* that when provided with a threshold minimum number of movies, lists all **pairs** of cast members that are featured together in at least minMovies along with the titles of such movies. For instance, coCast(5) will produce a list of pairs of cast members which are featured together in at least 5 movies.
14. In particular, run the method for minMovies =13 and produce the listing. It should give you nine pairs of cast members. Your method should not take more than 2 to 3 minutes to run. You must meet the runtime performance.
15. Also find **the** pair of cast members who have featured in the most number of movies together. Yes, there is only one and produce the movie title along with the names of the two cast members. Of course, if you solved question 15 above, you got this one as well.