Programming Project of Data Structures and Algorithms: Managing a Social Network

Preface

The aim of this programming project is to advance in the study of the design and implementation of data structures and algorithms throughout the course. The development of the project will require, most probably, to rethink program designs and programming alternatives previously made, but those activities are considered part of course goals. All the design decisions made, the project management and the program code will be documented in a *Project Report* which will record all activities and tasks performed. The report will be used to assess the task completed at any time. The Project Report will be structured according to a specification described in another document. Whenever possible use Junit for testing your code.

Requirements and project description

The software application to be developed will have to manage a social network. This social network is formed by people that may be linked among each other if there is a friendship relationship among them. For each person the following items will be recorded:

- identifier (unique)
- name
- surname(s)
- birth date
- gender
- birthplace
- home
- studiedat (he/she could have studied at many places)
- workedat (he/she could have worked at many places)
- movies
- groupcode

New people can be added to the network or deleted from it, besides other actions that may be of interest for the project.
At least, the following actions must be implemented:

1. The application must present an initial menu with the different choices for interacting with the social network.
   For example
   ```
   "MY_MENU"
   1. Load 'people' into the network ...
   2. Load 'relationships'...
   3. Print out people
   4. Search ...
   ...
   ...
   n. Log out
   ```
2. On the social network: take the data from a person and add him/her to the network. This function does not read any data from Console. This function receives the data as parameters and adds that information to the network.
3. On the social network: upload people's data (*people.txt*) from a file in your directory and add them to the social network. Use the function developed in the previous point. Your software application must be able to load many files, one at a time.
   The file ***people.txt*** *(people1.txt, peopleNice.txt,..., peopleFuN.txt, etc)* describes the collection of people in the network. Only the people appearing in that file are considered to belong to the social network. Each person identifier has only one occurrence in the file, but the information for that person may be incomplete. Only the identifier is compulsory.
   Your software application must be able to load several "people" files (one at a time).
4. On the social network: print out a listing to a text file of the people on the network.
5. The file named ***friends.txt*** *(friendsFun.txt, friendsClique1.txt, etc.)* (see below) has in each row two person identifiers that represent a friendship relation. The task is to implement an operation that relates them in the network, that is, the relationship has to be stored in the network.
   Example of the content of the file *friends.txt* (note: friendship relations are reciprocal).
   ```
   Jon232,Mikel34
   ```

```
Mikel34,Xabi112
Roberto22,Xabi112
Miriam21,Xabi112
Leire1,Miriam21
...
```

The file friends.txt describes the friendship relation among people as pairs of identifiers. A pair is not taken into account if either of its identifiers does not appear in the file people.txt. If an identifier appears in the file people.txt but it is not mentioned in the file friends.txt, that person does not have friends in the social network.

1st Milestone will assess the previous points

6. On the social network: given a person surname, retrieve his/her friends. If there are several people with the same surname then you should print out the list of friends for each of those people. The information to be retrieved can be printed out in the console or to a file (Person's id and Surname).
7. On the social network: given a city, retrieve all people who were born there. Information to be retrieved for each person is Person's id and Surname.
8. On the social network: retrieve the people who were born between dates D1 and D2, sorted by *birthplace, surname, name*. We consider only the "year" of the date, disregarding the values of month and day. The order relationship will be implemented according to the lexicographic order (dictionary's order) of the strings used for the attributes.
9. Given a set of identifiers in a file named ***residential.txt***, recover the values of the attributes *name, surname, birthplace* and *studiedat* of the people on the network whose *birthplace* matches the *hometown* of the people who are described in *residential.txt*. People whose *birthplace/hometown* is unknown do not affect the result of this operation. For example, if the file *residential.txt* contains the identifiers Mike222 and Mary123, your task is to retrieve the hometown of Mike222 and Mary123 people, and find all people who were born in those towns.
10. Two users have the same profile if they match the same collection of favorite movies. Your task is to split the users into classes with the same profile and to build a

list of those classes.

2nd Milestone

11. Six degrees of separation is the theory that everyone on Earth is six or fewer steps away, by way of introduction, from any other person in the world, so that a chain of “a friend of a friend" statements can be made to connect any two people in a maximum of six steps (or five intermediaries).
    On the social network: given two people of the network, your task is to retrieve the shortest chain that relates them.
12. On the social network: given two people, recover the largest chain of different people linking them (duplicate intermediaries are not allowed). Use backtracking.
13. On the social network: retrieve all the cliques of friends (crews) with more than 4 friends. A clique is a group of friends in which each person has friendship with each other. Use backtracking.

Final review