CS 245 - Assignment 3 Fall, 2017

Assignment 3 - Six Degrees of Kevin Bacon

The goal of this assignment is to demonstrate your general use of data structures and particular mastery of Graphs and their algorithms. This will be accomplished by reading and manipulating data about actors[1] and the possible associations by the movies they have acted in.

Background

Six Degrees of Kevin Bacon is a knowledge game based on the "small world" or "six degrees of separation" concept. It is possible for a large network to be linked by a limited number of steps through one or more who are well-connected. As it turns out, Kevin Bacon may be less prolific than others — Meryl Streep or Ben Kingsley, for example — so this game may belie the biases of the inventors.

In this assignment, your implementation will read some movie credit data. Once read, your implementation will allow the user to search for pairs of actors and connect them using one of the shortest possible paths between pairs.

Requirements

Your implementation will read data from a file representing actors, allow the user to search this data for individual actors and find the possible connections between pairs of actors through the movies in which they have acted.

Requirement 1:. Your implementation must read the file `tmdb_5000_credits.csv`, which is available for download through the Kaggle's TMDB 5000 Movie Dataset[2]: https://www.kaggle.com/tmdb/tmdb-movie-metadata#. According to the Kaggle site, this data is more accurate than what is available through IMDB. In order to download this file, you will need credentials for the Kaggle site. If you are unable to get access to the 38MB file, please let the instructor know as soon as possible.

In you implementation, you must expect the user to provide the location of this file on command line. For example, if your program is called A3.java, the user is expected to launch your implementation with:

```
java A3 /tmp/data_files/tmdb_5000_credits.csv
```

---

[1] Here, I use "actors" in a gender neutral sense — i.e. this refers both to female and male actors.
[2] By my calculation, this dataset contains fewer than 5000 movies. However, it does contain about 100,000 actor records.

The `tmdb_5000_credits.csv` file is officially a CSV file; however, two of the fields (cast, crew) are in JSON format. As such, it may be useful to use a JSON parser such as the one found in JSON-simple. If your implementation uses a JSON parser other than JSON-simple, you must indicate so in your submission comments along with instructions on how to install the package you use.

Requirement 2: Your implementation must find one of the shortest paths between pairs of actors. The user is expected to provide actors by their names. If the actors' names are not found, your implementation must indicate so. When a valid pair of actors are found, your implementation must find one of the shortest paths between the pair. If no such path exists, your implementation must indicate this. Here is an example, with the output of the implementation in bold:

```
Actor 1 name: David Guy Brizan
No such actor.
Actor 1 name: Hailee Steinfeld
Actor 2 name: Abigail Breslin
Path between Hailee Steinfeld and Abigail Breslin: Hailee Steinfeld -->
Abigail Breslin
Actor 1 name: Asa Butterfield
Actor 2 name: Paul Dano
Path between Asa Butterfield and Paul Dano: Asa Butterfield --> Abigail
Breslin --> Paul Dano
```

Extra Credit

This assignment has two extra credit opportunities. For these two opportunities, you may pick from the three options below. Any extra credit is added to the score for your second midterm grade. The addition to your midterm score is calculated a 5% improvement to the difference between the midterm grade and a perfect score. For example, if your grade on the midterm is 75, the value of each correctly-implemented opportunity is 5% x (100 - 75) = 5% * 25 = 1.25.

Opportunity 1: Standardise the search. By default, the names of actors are case sensitive. For example, a search for actor "Benicio Del Toro" will yield no results by default although there are two dozen records of his work in the data file.

Opportunity 2: Create a visual for the actor network. Your implementation may use a package such as JUNG or another visualisation package. Regardless of the package you use, you must indicate which package(s) you use in your submission comments along with instructions on how to install the package(s). Your implementation may ONLY use the additional package(s) for visualisation, not to satisfy any of the requirements in the previous section.

Opportunity 3: Design your own opportunity. With the written approval of the instructor, you may design your own extra credit opportunity to extend this assignment.

Submission

Submit the source code for your implementation on Canvas. You may also add any comments in a README file (text, PDF or Word document) to help the grader understand or execute your implementation. ~~If you include any~~

If you submit something in compliance with an extra credit opportunity, you must add a file entitled README_EXTRA_CREDIT (text, PDF or Word document) giving the details of your implementation.

Grading

Your grade for this assignment will be determined as follows:

- 55% = Implementation: your class implementations must run successfully with the source files and data provided. It must produce the expected results, a sample of which appears in the Implementation section above. Any deviation from the expected results results in 0 credit for implementation.
- 20% = Decomposition: in the eyes of the grader, your solution follow the suggestions above or otherwise must represent a reasonable object-oriented and procedural decomposition to this problem.
- 15% = Efficiency: your code must consistently use the most efficient data structures for the task and must consistently use the most efficient algorithms on those data structures.
- 10% = Style: your code must be readable to the point of being self-documenting; in other words, it must have consistent comments describing the purpose of each class and the purpose of each function within a class. Names for variables and functions must be descriptive, and any code which is not straightforward or is in any way difficult to understand must be described with comments.

Late assignments will be accepted with a 10% penalty per day