Programming Exercise: Step One

In this exercise, you will use the provided classes **Movie.java**, **Rating.java**, and **Rater.java** and read in and store information about movies and ratings of movies by different movie raters to answer simple questions about both movies and ratings.

For this assignment you will be given three starter files.

- The class **Movie** is a Plain Old Java Object (POJO) class for storing the data about one movie. It includes the following items:
    - Eight private variables to represent information about a movie including:
        - **id** - a String variable representing the IMDB ID of the movie
        - **title** - a String variable for the movie's title
        - **year** - an integer representing the year
        - **genres** - one String of one or more genres separated by commas
        - **director** - one String of one or more directors of the movie separated by commas
        - **country** - one String of one or more countries the film was made in, separated by commas
        - **minutes** - an integer for the length of the movie
        - **poster** - a String that is a link to an image of the movie poster if one exists, or "N/A" if no poster exists
    - A constructor with eight parameters to initialize the private variables
    - Eight getter methods to return the private information such as the method **getGenres** that returns a String of all the genres for this movie.
    - A **toString** method for representing movie information as a String so it can easily be printed
- The class **Rating** is also a POJO class for storing the data about one rating of an item. It includes
    - Two private variables to represent information about a rating
        - **item** - a String description of the item being rated (for this assignment you should use the IMDB ID of the movie being rated)
        - **value** - a double of the actual rating

  - A constructor with two parameters to initialize the private variables
  - Two getter methods **getItem** and **getValue**
  - A **toString** method to represent rating information as a String
  - A **compareTo** method to compare this rating with another rating
- The class **Rater** keeps track of one rater and all their ratings. This class includes:
  - Two private variables
    - **myID** - a unique String ID for this rater
    - **myRatings** - an ArrayList of Ratings
  - A constructor with one parameter of the ID for the rater
  - A method **addRating** that has two parameters, a String named **item** and a double named **rating**. A new Rating is created and added to **myRatings**
  - A method **getID** with no parameters to get the ID of the rater
  - A method **hasRating** that has one parameter **item**. This method returns true if this item is in myRatings, and false otherwise.
  - A method **getRating** that has one parameter **item**. This method returns the double rating of this item if it is in **myRatings**. Otherwise this method returns -1
  - A method **numRatings** that returns the number of ratings this rater has
  - A method **getItemsRated** that has no parameters. This method returns an ArrayList of Strings representing a list of all the items that have been rated.

Data files

You'll use several data files for this project. There are multiple versions of each type of file or different sizes. In creating recommendations you'll need two data files: one of movies, and one of ratings of these movies by different raters.

First there is a CSV file with movie data, the smallest of which is called **ratedmovies_short.csv**. The other similar files are larger, so we have created this small file with just a few entries in it that you can use for testing. Each file of this type has a header row as the first line, first followed by one line for each movie. Shown below is this shorter file that has six lines, the header line first followed by six movies. The lines are so long that they are wrapping in this document, each movie line is displayed here in two to three lines. For example the first movie has the IMDB number 0006414, the name of the movie is "Behind the Screen," and the movie was made in 1916 in the USA. The genres for this movie are Short, Comedy, and Romance. The director is Charles Chaplin, and the length of the film is 30 minutes. The last item is a link to a .jpg image of a poster for the movie if one exists, or "N/A" if there is not one.

ratedmovies_short.csv:

```
id,title,year,country,genre,director,minutes,poster
0006414,"Behind the Screen",1916,"USA","Short, Comedy, Romance","Charles
Chaplin",30,"http://ia.media-imdb.com/images/M/MV5BMTkyNDYyNTczNF5BMl5Ban
BnXkFtZTgwMDU2MzAwMzE@._V1_SX300.jpg"
0068646,"The Godfather",1972,"USA","Crime, Drama","Francis Ford
Coppola",175,"http://ia.media-imdb.com/images/M/MV5BMjEyMjcyNDI4MF5BMl5Ba
nBnXkFtZTcwMDA5Mzg3OA@@._V1_SX300.jpg"
0113277,"Heat",1995,"USA","Action, Crime, Drama","Michael
Mann",170,"http://ia.media-imdb.com/images/M/MV5BMTM1NDc4ODkxNV5BMl5BanBn
XkFtZTcwNTI4ODE3MQ@@._V1_SX300.jpg"
1798709,"Her",2013,"USA","Drama, Romance, Sci-Fi","Spike
Jonze",126,"http://ia.media-imdb.com/images/M/MV5BMjA1Nzk0OTM2OF5BMl5BanB
nXkFtZTgwNjU2NjEwMDE@._V1_SX300.jpg"
0790636,"Dallas Buyers Club",2013,"USA","Biography, Drama","Jean-Marc
Vallée",117,"N/A"
```

Next there is a CSV file with rating/rater data, the smallest of which is called **ratings_short.csv**. Again, the other similar files are quite large so we have created this small file with just a few entries in it that you can use for testing. Each file of this type has a header first followed by one line for each rating. Shown below is this shorter file that has eleven lines. The first line is the header. The second line shows the rater_id is 1, the IMDB movie ID is 0068646, the movie was rated a 10, and the time was 1381620027.

ratings_short.csv:

```
rater_id,movie_id,rating,time
 1,0068646,10,1381620027
 1,0113277,10,1379466669
 2,1798709,10,1389948338
 2,0790636,7,1389963947
 2,0068646,9,1382460093
 3,1798709,9,1388641438
 4,0068646,8,1362440416
 4,1798709,6,1398043318
 5,0068646,9,1364834910
 5,1798709,8,1404338202
```

Assignment

In this assignment you will create a new class named **FirstRatings** to process the movie and ratings data and to answer questions about them. You may find it helpful to use CSVParser and CSVRecord. Note that FirstRatings will need the following three import statements:

```
import edu.duke.*;
import java.util.*;
import org.apache.commons.csv.*;
```

Specifically for this assignment you will write the following methods in a new class named FirstRatings:

- Write a method named **loadMovies** that has one parameter, a String named **filename**. This method should process every record from the CSV file whose name is filename, a file of movie information, and return an ArrayList of type Movie with all of the movie data from the file.
- Write a void method named **testLoadMovies** that should do several things.
    - Call the method **loadMovies** on the file **ratedmovies_short.csv** and store the result in an ArrayList local variable . Print the number of movies, and print each movie. You should see there are five movies in this file, which are all shown above. After this works you should comment out the printing of the movies. If you run your program on the file **ratedmoviesfull.csv**, you should see there are 3143 movies in the file.
    - Add code to determine how many movies include the Comedy genre. In the file **ratedmovies_short.csv**, there is only one.
    - Add code to determine how many movies are greater than 150 minutes in length. In the file **ratedmovies_short.csv**, there are two.
    - Add code to determine the maximum number of movies by any director, and who the directors are that directed that many movies. Remember that some movies may have more than one director. In the file **ratedmovies_short.csv** the maximum number of movies by any director is one, and there are five directors that directed one such movie.
- In the FirstRatings class, write a method named **loadRaters** that has one parameter named **filename**. This method should process every record from the CSV file whose

name is filename, a file of raters and their ratings, and return an ArrayList of type Rater with all the rater data from the file.

- Write a void method named **testLoadRaters** that should do several things.
    - Call the method **loadRaters** on the file **ratings_short.csv** and store the result in a local ArrayList variable . Print the total number of raters. Then for each rater, print the rater's ID and the number of ratings they made on one line, followed by each rating (both the movie ID and the rating given) on a separate line. If you run your program on the file **ratings_short.csv** you will see there are five raters. After it looks like it works, you may want to comment out the printing of each rater and their ratings. If you run your program on the file **ratings.csv**, you should get 1048 raters.
    - Add code to find the number of ratings for a particular rater you specify in your code. For example, if you run this code on the rater whose rater_id is 2 for the file **ratings_short.csv**, you will see they have three ratings.
    - Add code to find the maximum number of ratings by any rater. Determine how many raters have this maximum number of ratings and who those raters are. If you run this code on the file **ratings_short.csv**, you will see rater 2 has three ratings, the maximum number of ratings of all the raters, and that there is only one rater with three ratings.
    - Add code to find the number of ratings a particular movie has. If you run this code on the file **ratings_short.csv** for the movie "1798709", you will see it was rated by four raters.
    - Add code to determine how many different movies have been rated by all these raters. If you run this code on the file **ratings_short.csv**, you will see there were four movies rated.