Programming Exercise: Step Two

In this exercise, you will add to the programming project you did for the first assignment. You will continue to use the provided classes Movie.java, Rating.java, and Rater.java that were provided for the first assignment and you will also use the FirstRatings class you wrote to read in and store information about movies and ratings of movies by different movie raters. You will build on this assignment by calculating average ratings of movies. You will work with two new classes, **SecondRatings** which we will give you parts of, and a new class **MovieRunnerAverage**, which you will create.

Assignment

In this assignment you will modify a new class named **SecondRatings**, which has been started for you, to do many of the calculations focusing on computing averages on movie ratings. You will also create a second new class named **MovieRunnerAverage**, which you will use to test the methods you created in SecondRatings by creating a SecondRatings object in MovieRunnerAverage and calling its methods.

Specifically for this assignment you will write the following classes and methods:

- Note that the SecondRatings class has been started for you. This class includes two private variables, one named **myMovies** of type ArrayList of type Movie, and a second one named **myRaters** of type ArrayList of type Rater. A default constructor has also been created for you. Until you create the second constructor (see below), the class will not compile.
- Write an additional SecondsRating constructor that has two String parameters named **moviefile** and **ratingsfile**. The constructor should create a FirstRatings object and then call the **loadMovies** and **loadRaters** methods in FirstRatings to read in all the movie

and ratings data and store them in the two private ArrayList variables of the SecondRatings class, **myMovies** and **myRaters**.

- In the SecondRatings class, write a public method named **getMovieSize**, which returns the number of movies that were read in and stored in the ArrayList of type Movie.
- In the SecondRatings class, write a public method named **getRaterSize**, which returns the number of raters that were read in and stored in the ArrayList of type Rater.
- Create a new class named MovieRunnerAverage. In this class, create a void method named **printAverageRatings** that has no parameters. This method should:
    - Create a SecondRatings object and use the CSV filenames of movie information and ratings information from the first assignment when calling the constructor.
    - Print the number of movies and number of raters from the two files by calling the appropriate methods in the SecondRatings class. Test your program to make sure it is reading in all the data from the two files. For example, if you run your program on the files **ratings_short.csv** and **ratedmovies_short.csv**, you should see 5 raters and 5 movies.
    - We will add more code to this method in a bit.
- In the SecondRatings class, write a private helper method named **getAverageByID** that has two parameters: a String named **id** representing a movie ID and an integer named **minimalRaters**. This method returns a double representing the average movie rating for this ID if there are at least **minimalRaters** ratings. If there are not **minimalRaters** ratings, then it returns 0.0.
- In the SecondRatings class, write a public method named **getAverageRatings**, which has one int parameter named **minimalRaters**. This method should find the average rating for every movie that has been rated by at least **minimalRaters** raters. Store each such rating in a Rating object in which the movie ID and the average rating are used in creating the Rating object. The method **getAverageRatings** should return an ArrayList of all the Rating objects for movies that have at least the minimal number of raters

supplying a rating. For example, if **minimalRaters** has the value 10, then this method returns rating information (the movie ID and its average rating) for each movie that has at least 10 ratings. You should consider calling the private **getAverageByID** method.

- In the SecondRatings class, write a method named **getTitle** that has one String parameter named **id**, representing the ID of a movie. This method returns the title of the movie with that ID. If the movie ID does not exist, then this method should return a String indicating the ID was not found.
- In the MovieRunnerAverage class in the **printAverageRatings** method, add code to print a list of movies and their average ratings, for all those movies that have at least a specified number of ratings, sorted by averages. Specifically, this method should print the list of movies, one movie per line (print its rating first, followed by its title) in sorted order by ratings, lowest rating to highest rating. For example, if **printAverageRatings** is called on the files **ratings_short.csv** and **ratedmovies_short.csv** with the argument 3, then the output will display two movies:

```
8.25 Her
9.0 The Godfather
```

- In the SecondRatings class, write a method **getID** that has one String parameter named **title** representing the title of a movie. This method returns the movie ID of this movie. If the title is not found, return an appropriate message such as “NO SUCH TITLE.” Note that the movie title must be spelled exactly as it appears in the movie data files.
- In the MovieRunnerAverage class, write the void method **getAverageRatingOneMovie**, which has no parameters. This method should first create a SecondRatings object, reading in data from the movie and ratings data files. Then this method should print out the average ratings for a specific movie title, such as the movie “The Godfather”. If the **moviefile** is set to the file named **ratedmovies_short.csv**, and the **ratingsfile** is set to the file **ratings_short.csv**, then the average for the movie “The Godfather” would be 9.0.