ECE 325 LAB Assignment 4: Processing audience feedback

Now that you have played some gigs, you can use audience feedback to help you decide which song and instrument combinations are great and which ones aren't. For every song you play with a certain set of instruments, you recorded the following in a text file:

- The Song title
- The list of instruments used to play it
- The audience rating (between 0 and 10 with 10 being the best)

Because audiences differ, you have multiple ratings for some songs. You decide to revisit the SongCollection that you created earlier (actually, you decide to write it from scratch, so no need to look at the old code). Instead of Strings, the SongCollection will now hold Song objects (that have a title, a list of instruments used to play it and a rating).

Part 1: Implement the application

Finish the implementation to load the SongCollection (including the average ratings) from the songratings.txt file in the provided classes.

A Song should occur only once in the SongCollection – make sure to update the average rating when you encounter multiple ratings for the same song.
So, for example:

```
Contribution;Guitar,Guitar,Drums;8
Contribution;Guitar,Guitar,Drums;10
```

Should be stored in the SongCollection as a Song with title “Contribution”, instruments “Guitar”, “Guitar” and “Drums” and an average rating of 9.

Hints:
- SongCollection uses an ArrayList internally to hold the Song objects.
- You will need to override the equals() method in the Song class. For now you can assume that two Song objects are equal iff their titles and list of instruments are the same.
- To make your life easier, you can treat the list of instruments as an ArrayList of String objects. So e.g. “Guitar,Guitar,Mic” should result in an ArrayList with the Strings “Guitar”, “Guitar” and “Mic”. The order of the instruments has no meaning, so “Guitar,Guitar,Mic” should be considered the same as “Guitar,Mic,Guitar” by your application. Note that a song can be played with different combinations of instruments!
- The equals method(s) in the Song class are quite important for the correct functioning of the application. Make sure that they do not crash on unexpected input.
- The Collections class in the Java API contains some functionality that can helpful for your application.

Part 2: Questions about design choices

Please answer these questions about the implementation/design choices. There is no need to change your implementation based on these questions, a textual explanation is fine.

1) The Song class contains two equals() methods. Explain the difference using your knowledge about inheritance.

2) Why do we need the version of equals() that takes an Object as the parameter in this class?

Rubric

Implementation of the application 16 points total.

Questions 2 points each.

(20 points total)

Please submit:
1) A zip file containing your code and a PDF with the answers to the questions above.
Name the file 'FirstName_ID_lab_asg4.zip' and keep the exact same file structure as the zip that was provided for the assignment. So, for example:
Filename: Cor-Paul_1234567_lab_asg4.zip
|------- solution.pdf
|------- src
| |------- ece325
| | |------- labs
| | | |------- lab4
| | | | |------- *.java

2) A screencast/movie that shows the following steps:

- Open your eClass with your name shown
- Open your IDE
- Show your code briefly
- Execute your code and demonstrate that your class works correctly.

Please submit the screencast as a **separate** file to eClass.

Please do not modify any of the names/methods we've defined in the provided *.java files.