Lab4 documentation

1. Resources

For this lab I imported the CSV files from Spotify Charts. For the weeks of 10/15/2020 to 10/21/2020; 10/08/2020 to 10/14/2020 and 10/01/2020 to 10/07/2020 in the United States. I took off the header.

2. Prerequisites

In order to be able to use this program efficiently, the user must use a csv file (without the header) that follows the chart:

Position	Track name	artist	streams
….	….	….	…..

The user must have four files (*week1.csv, week2.csv, week3.csv* and exampleOutput.txt)) located in the same directory as the Lab4.java file.

3. Implementation

Problem description:
Your VIP client wants to listen to the music tracks in this week's latest song list to review them and get a sense for what they sound like. This time your client wants you to process all the full weeks of the fiscal quarter that just passed. This means you're working with multiple CSV files of the same format.
Your client also says they want to listen to songs based on song track titles in ascending order. It "must be in this order" they say.

How I wrote the program:

First, I wrote a class Song which takes a track name and an address of the next song as instances. Then, I created a class Playlist (linked list) that represent a playlist, which takes a Song First (the address of the first element of the Linked list) and which has some methods such as:

- ***addSong(String track):*** to add a song in the play list
- ***ListenToSong() :*** retrieves the next song to listen to. " >> "
- **isEmpty():** to check whether the list is empty or not
- **find(String s) :** to check if the name "s" already exist
- ***sortList():*** to sort the elements of the list by ascendant order (A.....Z)
- ***display(PrintStream ps):*** to print the play list songs inside the output file (exampleOutput.txt)

Second, I created a class MyQueue which inherits from Playlist, it creates a linked list that stores songs from an array of text file. Then, I wrote a function called Filter which takes a string (a line of the csv file) and then filter it (the line) to get only the track name then return it.

Finally, I created a class SongHistoryList (stack) that represent the song history. After a song has been listen (for each ListenToSong()), it is right the way play in the songHistoryList where it represents the " << " (previous song)

Hint: There is no redundancy inside MyQueue (it stores a track only once).