U4B Lab - ArrayLists
Spotify Lab (2026) (video demo)

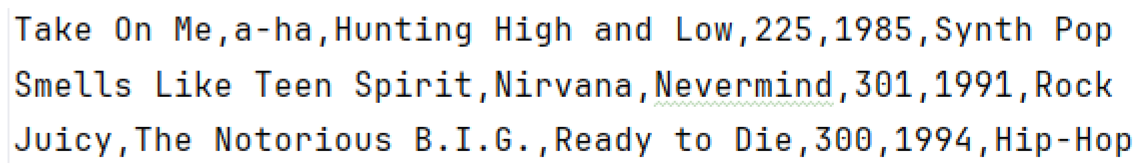
In this lab you will create a user interface for a playlist of songs on Spotify.

- First, you will read in a text file containing song data for each track on the playlist.
- Then you will give the user a menu of choices to choose from regarding the playlist.
- Depending upon what they choose, you will display the requested data.
- The user will be able to interact with the menu until they choose the option to quit.

This dataset is a curated educational snapshot inspired by Spotify's and is not guaranteed to reflect live rankings. Here is a link to the txt file.

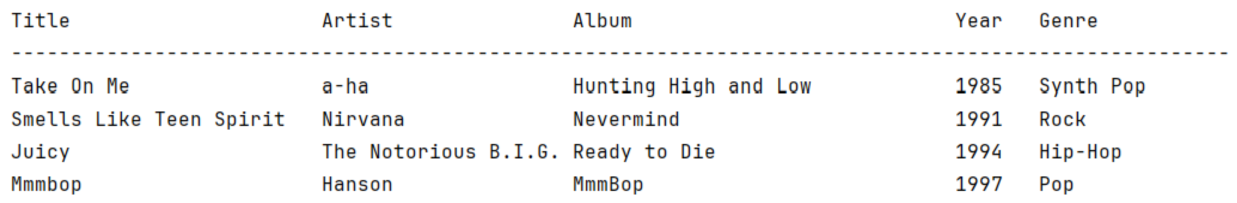
The text file will be in this format:

title, artist, album, durationSeconds, releaseYear, genre



```
Take On Me,a-ha,Hunting High and Low,225,1985,Synth Pop
Smells Like Teen Spirit,Nirvana,Nevermind,301,1991,Rock
Juicy,The Notorious B.I.G.,Ready to Die,300,1994,Hip-Hop
```

The data displayed on the screen to the user should be in this format:



```
Title                     Artist               Album                            Year   Genre
-------------------------------------------------------------------------------------------------------
Take On Me                a-ha                 Hunting High and Low             1985   Synth Pop
Smells Like Teen Spirit   Nirvana              Nevermind                        1991   Rock
Juicy                     The Notorious B.I.G. Ready to Die                     1994   Hip-Hop
Mmmbop                    Hanson               MmmBop                           1997   Pop
```

To correctly complete this assignment, you will need to create three classes as described below that provide the functionality shown in the demo. The data for each song needs to be displayed on one line (as shown above).

- **Create a `Song` class**
  - Constructor
  - toString for one Song (in a row format) as demonstrated above
  - get/set methods as needed

- **Create a `Playlist` class that has an ArrayList of `Song` objects**
  - Constructor
  - toString that will return the display showing the entire ArrayList of Songs (as shown above with correct spacing/columns lined up). To assist with this, use the String.format() method.
  - Several helper methods. Create parameters as needed. Return type listed after method description.
    - Method to read in Song data (void)
    - Method to Search by and display all Songs in a specific `genre` (void)
    - Sort by `artist` (Selection sort) (void)
    - Sort by `releaseYear` (Insertion sort) (void)
- **`SpotifyTester`**
  - Show menu of choices and user selects a choice to sort by Artist A-Z, Z-A, sort by releaseYear old-new or new-old, search by genre, or quit.
  - If they don't enter a valid option 1-7, then they are prompted to enter a correct option (you are preventing bad user input and must use a try-catch block)
  - **I should not be able to crash your program by entering bad input data**
  - Use a while loop to allow the user to continually interact with the menu until they choose to quit.
  - Create a helper method that can be used to validate this input for menu choices
  - Create constants that are equal to the int values 1-7 so that your code is more readable, rather than using the values 1-7 in your if statements.

- Other items…
  - Your entire lab should only have **ONE Scanner** for the keyboard. If you need to use this Scanner in other methods, then pass it as a reference in a parameter.
  - When you are done using a Scanner object, close the Scanner
  - Use a try/catch block to avoid crashing the program

We will have checkpoints along the way to help you stay accountable

- Create the three classes, have the method to read in data, and call this method in Driver. Print the Playlist object to demonstrate both toString methods look as desired.

- Create a loop in the driver, show the menu of choices, get and validate user input. You must use try/catch blocks to ensure user input doesn't cause an exception!!!! If the user enters a bad value, they should be notified of the error, and allowed to continue with the corrected input.

- Write the Linear Search code to display **all** shows in a specified genre. If there are no shows, then display a nice message saying so.

- Finish the lab by creating the sortByArtist method that uses Selection sort to sort alphabetically and the sortByRelease method that uses Insertion sort to sort numerically.

- **GITHUB:**
    - You are expected to commit your code to the U4B GitHub Classroom assignment at the end of class each day. Not only does this protect you from losing your work but it also **verifies the authenticity of your work and the natural progression of your project as you make progress each day.**

    - You are expected to make appropriate commit messages such as "completed user menu" or "insertion sort still isn't working correctly". We do NOT want to see messages like "end of class" or "asdf" etc.

    - This is NOT a partner project.

    - You may NOT use AI for the completion of this project