CISC 1115 Homework 10

This homework will give you practice reading in data, writing to files, writing and calling methods, and creating, populating, printing, sorting, searching, and modifying parallel arrays.

Write a Java program to create a database of films. Our database has been compiled as a class.

First, use the classList.txt file already made for you here. (Download and import it, copy it into a new text file with the title empireList, etc. Do it however you see fit)

- In the main method, create four parallel arrays to hold each show's: startYear, title, countryOfOrigin, and endYear (this may be ONGOING) respectively.

- Create a method called **readData()** to populate the arrays from the empireList.txt file. The data is in the order of ***startYear, title, countryOfOrigin, endYear*** (each piece of data is separated by a comma (',') Hint: the String class has a method called split() that splits a given string around matches of the given regular expression, how can we use this to read each piece of data into a separate array?)

- Create a method called **printData()** that will print a header and the data from the arrays to an output file in a four-column table format: **startYear, title, countryOfOrigin, endYear**

- Use the selection sort algorithm to sort the data in the following way:

  - Create a method called **sortByYear()** to sort the data in descending order by startYear.

- Call the **printData()** method to print to file, each with a specific header of how the data is sorted (e.g. "Sorted by Premiere Year:")

- Use the bubble sort algorithm to sort the data in the following way:

  - Create a method called **sortByTitle()** to sort the data in alphabetical order (A-Z) by title

- Call the **printData()** method again to print to file, each with a specific header of how the data is sorted (e.g. “Sorted by Title (A-Z):”)

- Once sorted alphabetically, use the following method to create a new array that holds each title only once:

```
public static int removeDuplicateShows(int numberOfShows, String[] startYear, String[]
title, String[] countryOfOrigin, String[] endYears, String[] uniqueShows) {
    if (title.length == 0 || title.length == 1) {
        return title.length;
    }

    int uniqueCount = 0;
    boolean[] isDuplicate = new boolean[title.length];

    for (int i = 0; i < numberOfShows; i++) {
        if (!isDuplicate[i]) {
            uniqueShows[uniqueCount++] = title[i];
            for (int j = i + 1; j < numberOfShows; j++) {
                if (!isDuplicate[j] && title[i].equals(title[j])) {
                    if (!countryOfOrigin[i].equals(countryOfOrigin[j])) {
                        uniqueShows[uniqueCount++] = title[j] + " (" + countryOfOrigin[j] + ")";
                    }
                    isDuplicate[j] = true;
                }
            }
        }
    }

    return uniqueCount;
  }
```

- Create a method called **mostPopular()** that will answer the following question: ***what is the most popular show?*** (which show(s) show up on the list the most frequently) and print to the output file. (If more than one show shows up the most, you only need to print the first one encountered -- hint: remember the code for determining the most frequent integer)

- Create methods for searching the data by title, country, or ongoing status. (hint: use a separate method for each)

    - search for show by title
    - search for all shows that have the same country of origin

    - search for all shows that are currently ongoing
        - if not found output error:

          "Unfortunately there is no record of that show in this list."
        - if found:

          Title (Year) Country

- Create a method called **addShow()** that prompts the user to add a new show to the data

    - prompt the user for the show that they would like to add
    - **the show cannot be already on the list BUT can be a remake from a different country** (so it CANNOT have the same title AND countryOfOrigin)
    - If the show is not already on the list:
        - add title, startYear, endYear (or ONGOING), and countryOfOrigin

The output file should be in the following order:

- The entire data set, ordered by year
- The entire data set, ordered by title
- What the most popular show is
- The show, country, and ongoing status searched for, and the results of each search
- The updated data set, ordered by title

Print everything except for prompts to an output file.

Submit the Java file and the output file.