Fall 2026 COP 3330 Program #9: Files

Jaz's Concert Attendance

Background Information

In preparing to make this assignment, I quickly searched the web for appropriate text files of data for music groups or concerts. I wasn't able to find anything interesting with my first few tries. At that point in our TA meeting, Jaz mentioned that she keeps a spreadsheet (.csv form) of every concert she's attended since 2019 and that she would be happy to donate that spreadsheet for the assignment.

Thus, the basis for this assignment will be processing the data in Jaz's spreadsheet that she so kindly donated so that the class could have some practice with files.

You'll be given the original .csv file as well as a very basic java program which opens the file, reads in every line of the file into an ArrayList (of String), and closes the file.

Each string is comma separated (that's what csv stands for, "comma separated values"), and the fields for each record are: Date, Band, Show/Tour, Venue and Location.

Assignment

For this assignment, you will write a program that will prompt the user to enter a file with the salary data. You must catch the FileNotFoundException and end the program if the user doesn't enter a valid file. (So no reprompting; the user gets one chance to get it right! But by catching the exception, you end the program gracefully, without crashing.)

Assuming the user enters a valid file (we'll only test your program with the file I've provided), you will then provide the user with the following menu of choices:

1. Output all data sorted by band, breaking ties by date. You will ask the user whether they prefer the output to the screen or to a file. If the answer is the latter, then ask the user which file the would like the output saved. The file you output should have a similar format to the input file: First row with the header labels separated by commas, followed by each entry on a single like, separated by commas. (A clever idea would just be to save the whole line from the original input...)

2. Ask the user to enter a band name and output, in chronological order, each show that Jaz has seen with that band. If Jaz never saw a show with the requested band, output a message to that effect.

3. **Anything interesting you can think of as a query a user might want to make of the data.**

4. **Anything else interesting you can think of as a query a user might want to make of the data.**

5. Quit

Implementation Requirements

Since the input file has spaces in some of the fields, you MUST READ all data, both from the file and from the user by line and then use a StringTokenizer, with a comma delimiter (not a space) to separate out different fields.

You are given some leeway in how you want to store the input data, but it is advised that you use some of the built-in data structures taught over the course of the previous two weeks.

When appropriate, do reasonable error checking. We've now learned multiple ways to error-check. Implement what you think is reasonable.

No sample input or output is provided, so you are to make reasonable decisions about the format. Please make sure you have appropriate documentation in your code so that the graders can easily interpret your intentions of how to use your program. The graders will test the actual functionality of your program only with the posted file, Jaz-ShowList.csv. (They will enter a bad file name to see if you catch the FileNotFoundException.) If something isn't specified here or is ambiguous here, use your best judgement to resolve the ambiguity. (Do not ask the course staff!)

Deliverables

Please submit two files:

1. **README.pdf** which contains a brief description of your program and how to use it, focusing on menu choices 3 and 4 which you have completely devised on your own.

2. **A zip file** with all of your source files. You do NOT need to include Jaz-ShowList-csv. The TAs will place a copy of this file in the directory with all of your Java files before they start grading your assignment.

Make sure to include a header comment for each file you submit and appropriate internal comments.