Spring 2026 COP 3330 Program #9: Files, Exceptions

UCF Salary Database

Background Information

The following website:

https://knightnews.com/2016/08/ucf-professor-salaries-released-by-gov-rick-scott-see-salaries-here/

lists the UCF faculty annual salaries from the 2016-7 academic year.

I have collated that information into a text file, "ucfsalaries-2016.txt" which has each field separated by tabs.

The first line of the file stores the number of faculty records and the faculty records follow, one per line.

Each line has four fields, each separated by tabs: last name, first name, position, and salary.

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 last name, breaking ties by first name to a file. You will ask the user to enter their desired filename for the output file. In addition, your program should print to the screen both the average and standard deviation of all faculty members.

2. Ask the user to enter a position, and a filename. Your program should write to the output file a sorted list of all faculty members that hold that position, sorted by last name, then first name. In addition, your program should print to the screen both the average and standard deviation of all faculty members who hold that particular position.

3. Ask the user to enter a last name, and you should output the records of all faculty members with that last name (output last, first, position and salary), in any order.

4. Ask the user to enter both a last name and a first name, then output the records of each faculty member who match both the first and last names entered.

5. Quit

For options 2, 3 and 4, create two separate Exceptions:

PositionNotFoundException
NameNotFoundException

The first should be thrown if the user enters a position that doesn't match any faculty member. (I think all of the positions listed are: PROFESSOR, ASSOCIATE PROFESSOR, ASSISTANT PROFESSOR, LECTURER, and INSTRUCTOR) But, in theory, your program should work with an arbitrary input file which has different positions listed than the given input file. **You will lose credit if you hard-code these particular positions.**

The second should be thrown either if no name in the set of records matches the last name entered OR if no name matches both the last and first names entered. Make the strings you generate in the Exception objects distinct for the two cases.

For each of these exceptions, catch them and handle the exception gracefully.

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 tab delimiter (not a space) to separate out different fields.

You must create a FacultyMember class, which implements Comparable<FacultyMember> and defines the sorted order based on last name, breaking ties by first name.

You must create a FacultyList class, which stores all of the relevant data from the input file. This class must have the following constructor:

```
public FacultyList(String inputFile) throws FileNotFoundException
{
    // Constructor code in here.
}
```

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.

In situations where you are given a query with no results, catch the corresponding Exception, print an appropriate error message, and reprompt the user with the menu without generating any other screen output or output file.

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, ucfsalaries-2016.txt. (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 three source files:

1. **FacultyMember.java** which contains only your code for the FacultyMember class.

2. **FacultyList.java** which contains only your code for the FacultyList class.

3. **FacultyData.java** which contains a main method (and possibly other methods) which runs your whole application. This main method should instantiate a single FacultyMember object.

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