Pre-Lab 2 Exercise

CS 106: Introduction to Data Structures

Note: this should be completed with your partner before starting Lab 2. You should begin this exercise inside the Lab 2 repository, after copying over your files from Lab 1. You should use a dedicated folder (cs106-lab2/src/main/java/prelab/) available in the lab2 starter code.

Goals:

- Get practice with creating classes, objects and reading in CSV data.

PRE-LAB

This exercise is a continuation from Pre-Lab 1 (with the StudentProfile.csv file). You should have at least gotten started on the String[] constructor last week. Please complete the constructor (and any other parts of that in-lab).

CREATING A DATA CLASS

We can think of the entire data as a collection of all of the rows. We can compile all the rows into a collection with a built-in object, ArrayList. When we read in the data, it will be stored as an ArrayList of String arrays. We will then convert it row by row into the data type you defined in Pre-Lab 1 (I will refer to this as **Student**, but you might have called it something else).

1. Create a class that holds the entire dataset (not just one row) of the CSV.
2. Make a variable of type ArrayList that holds the rows of objects (each of type **Student**).
3. Create a constructor that takes no input. In the constructor, initialize the instance variable into a new ArrayList.
4. The overall idea to populate your data structure is to iterate through the ArrayList of String arrays, convert each array into a **Student**, and add the converted **Student** into the ArrayList. Here are a couple suggestions about how to approach the design.
    a. Create a method that takes in an ArrayList of String arrays. In the method, you should use a loop to iterate through the input and add each converted **Student** into the data structure.
    b. Create a method that takes in a String array and adds the converted **Student** into your data structure. You could repeatedly call this method from a different place where you choose to iterate over the ArrayList of String arrays.

5. Override the toString method for this new class. It should use a for loop and call toString method for each **Student**, and build up the return value using StringBuilder. You can make use of the newline character i.e. "\n" to create a line break so that each student will appear on a separate line.

READING IN THE DATA

We will be using the Opencsv library to help us read in the csv file. The Opencsv library should already be listed as a dependency in the pom.xml file for you.

As mentioned above, the data read in will be of type ArrayList<String[]>. For example, the read-in data will exist like this (only the first two rows from the csv are shown):

```
ArrayList<String[]> dataReadRows =
<{"Will","20", "Sophomore", "Barclay","Yes","Michael Elias"} ,
{"Steve", "19", "First Year","Barclay", "No","Katrina Glanzer"} ,
… >
```

Let's read the csv data!

6. Inside the main method of the Main class, put in the following:

```
/* Header-aware reader of StudentProfile.csv
   Make sure StudentProfile.csv is accessible or provide the proper path
   Read and write to a file may throw an exception e.g., FileNotFound etc,
make sure you handle such errors properly */
CSVReaderHeaderAware csvReader = new
CSVReaderHeaderAware(new FileReader("StudentProfile.csv"));

// ArrayList containing each row of the csv as a String array
ArrayList<String[]> dataReadRows = new
ArrayList<String[]>(csvReader.readAll());

dataReadIn.close();
```

7. Try printing out the read-in data above to confirm what it looks like.
8. Correct any errors by adding any imports needed. Eclipse should help you with it. You can use a try/catch block to deal with exceptions related to reading in the file.

Now that you've read in the data, let's make an instance of the object you created earlier and pass in the data you read in!

9. Create an instance of the data collection class you made earlier in the main method.
10. Utilize the method you created in #4 to populate your dataset using your dataReadRows variable. Now, you should be able to use System.out.println(<instance of your object>) to see the entire list of **Student**s!