MIST352 Homework #3 (100 points)

# Problem statement:

You are tasked with developing a Java application that interacts with the employee\_data.txt file, containing employee information such as names, dates of birth, and roles. The application must use a Graphical User Interface (GUI) for operations like viewing employee details and searching for specific employees.

# Requirements:

1. Data Reading: The application should efficiently read from employee\_data.txt, enabling user interaction with the data.
2. GUI Interaction: Implement a GUI for viewing all employees, their roles, and for conducting searches by name or role.
3. Class Restriction: Use only the provided classes in the HW3 folder. You may add methods for testing, which should not be included in the final submission.
4. Documentation: Document your code with comments detailing each class and method's purpose and functionality.
5. Functionality: The program must be free from syntax errors. Non-functional programs will result in a significant deduction (up to 50% of the total grade).

# Submission Instructions:

At the start of each Java class file, include a comment with your name, course details, homework number, and a note on whether ChatGPT was used. If ChatGPT assisted, include the used prompt.

Deliverables:

Push your completed work to your Homework folder on your GitHub repository in the HW3 directory. Confirm your submission by posting a comment on eCampus with your GitHub repository link.

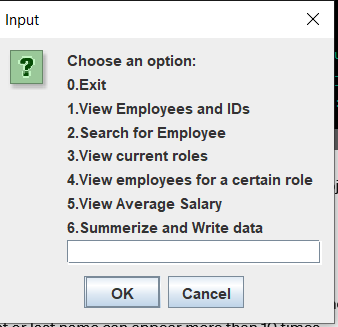
Note: You may consult ChatGPT for one Java class in the HW3 folder. Document any assistance received according to the guidelines.

# Instructions:

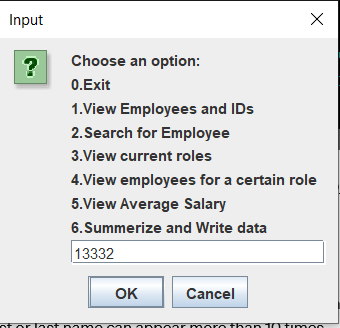
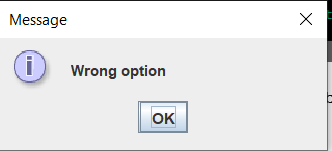
1. Pull the MIST352 repository and explore HW3 project under the Homeworks folder. Copy the HW3 project to your Homeworks folder of your local repository and open it in Eclipse.
   1. Only code the methods where the documentation states (as \* You need to code this).
   2. Do not modify any methods where the documentation states (\* Keep as is).
   3. Do not add your own methods. **[5 points]**
2. The DataSummary.java is a class that has two private variables, fileLocation and averageSalary. This class allows you to create an object of the text file that has the employee’s data. **[32 points]**
3. The Employee.java is a class that has 8 parameters as shown below. Do not add any setters or getters here, instead, initialize all parameters in the constructor.  **[32 points]**

|  |
| --- |
| **Employee** |
| -name: String |
| -dateOfBirth: String |
| -role: String |
| -employeeId: int |
| -email: String |
| -salary: double |
| -aboveAvg: boolean |
| -bar: String |
| +Employee(line: String) |
| +setAboveAvg(dblAverage: double): void |
| +SalaryBarChart(salary: double): String |
| +SummerizeEmployee(): String |

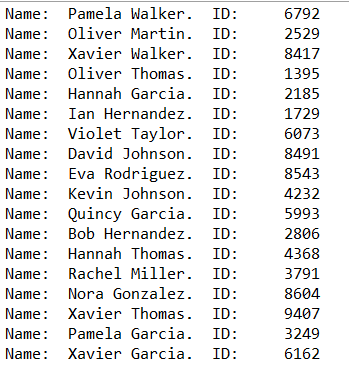
1. The HW3.java is the main method. Keep lines 26-35 as is and code a switch statement to evaluate the users input and inside the while loop. The switch statement should evaluate the number provided by the user, and then implement their request. All cases 1 through 5 will require you to use the object of type myData which is defined in line 34. For case 6, you need to read through the text file in the main method, and every time you read a line, you create an employee object using the line you just read. Run the program and see below: **[33 points]**



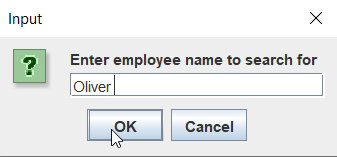
* 1. If the user type 0, then the program should exit, otherwise, the user should remain the in the program.
  2. If the user types of an invalid number, it should display wrong option as shown below:

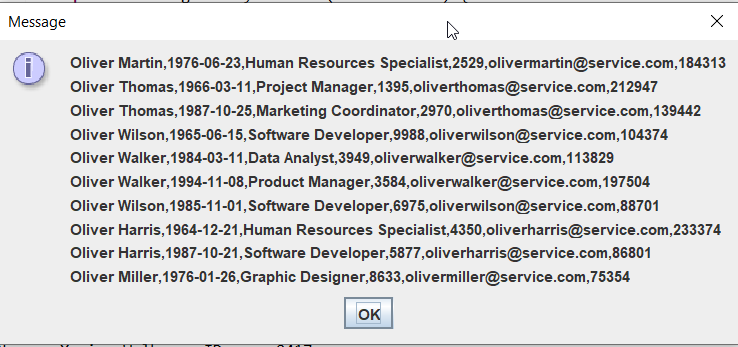
 

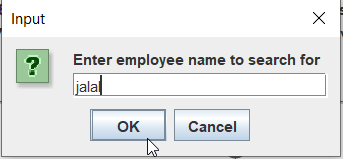
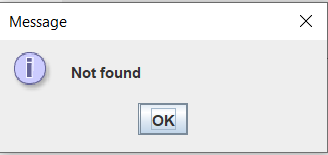
* 1. If the user types 1, then the program should display employees’ information to the **console** as shown below:



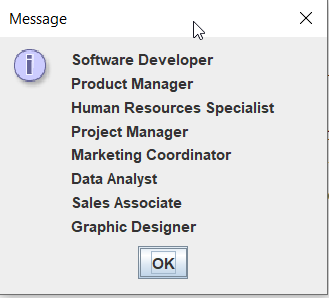
* 1. If the user types 2, then ask user for an employee name (first, last, or both) and display ALL employees’ information who have the same name. Otherwise, print not found as shown below.



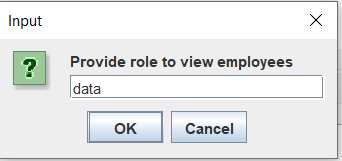


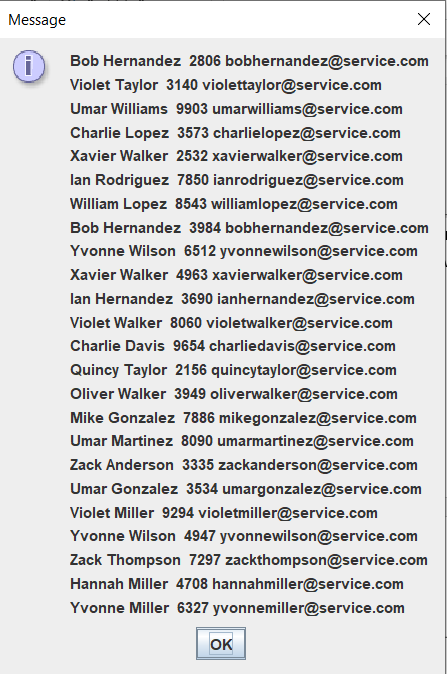
 

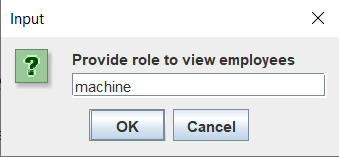
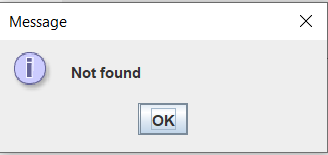
* 1. If the user types 3, then you should call the method (already coded for you) that displays the roles as shown below:



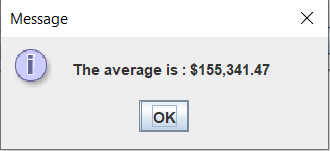
* 1. If the user types 4, then you should view all employees in that role exactly in the format below. If the role does not exist, then display not found as shown below.





* 1. If the user types 5, then display the average salary of all employees in the text file as shown below.



* 1. If the user types 6, then display to the console the names, IDs, the bar representation of the salary, and whether the salary is above or below the average. And write that data to the text file given to you in line 29 of the main method.

