Lab #2: Inheritance and Polymorphism in Java

A software company has two different kinds of employees: managers and technical staffs. Some managers are executives. Some technical staffs are software engineers and the rest are test engineers. Based on the description, complete the following tasks:

a. Identify the class hierarchy based on the underlined nouns above. Create a UML class diagram for this class hierarchy. You MUST find an online software for drawing UML. In UML diagram, you need to specify class name, class attributes and class methods.

Class	Attributes	Possible Operations
Employee	last name, first name, base salary, job title	- Calculate salary - Display the first and last name - Display base salary - Display job title - Display information (last name, first name, base salary, job title)
Manager	last name, first name, base salary, job title, bonus payment	- Calculate salary (base salary + bonus) - Display bonus payment - Display information (last name, first name, base salary, job title, bonus payment)
Executive	last name, first name, base salary, job title, bonus payment, stock option	- Calculate salary (base salary + bonus payment + stock option) - Display stock option - Display information (last name, first name, base salary, job title, bonus payment, stock option)
Technical staff	last name, first name, base salary, job title, profit sharing	- Calculate salary (base salary + profit sharing) - Display profit sharing - Display information (last name, first name, base salary, job title, profit sharing)
Software Engineer	last name, first name, base salary, job title, profit sharing, overtime pay	- Calculate salary (base salary + overtime pay) - Display overtime payment - Display information (last name, first name, base salary, job title, profit sharing. Overtime pay)
Test Engineer	last name, first name, base salary, job title, profit sharing	- Calculate salary (base salary + profit sharing) - Display information (last name, first name, base salary, job title, profit sharing)

b. Create Java classes based on the UML you created above and implement all the classes and member functions needed. Please note you need to implement the member functions in a way to allow for dynamic binding. A constructor is needed for each class. You must

make sure to use super class methods as much as you can when you implement sub class methods. All data members MUST be private. Implement necessary member functions if you need to access the private data members of super classes.

c. Write a testing class with a main method to test the classes defined above:

- In the main method, you first create an array of 3 elements of with type Employee.
- Then define three variables: the first variable is called emp0, the data type of it is Executive, the second variable is called emp1, the data type of it is Software Engineer, the third variable is called emp2, the data type of it Test Engineer. You should give the necessary information to create each object (i.e., needed by the constructor), for example, the following pass in the first name, last name, base salary, job title, bonus, profit sharing for an executive object.
  Executive emp0("John", "Doe", 100000, "VP", 1000000, 2000);

- Assign a value to each of the three array elements in the following order: emp[0] is assigned to the value of the Executive object (emp0), emp[1] is assigned to the value of the Software Engineer object (emp1) and emp[2] is assigned to the value of the Test Engineer object (emp2) respectively.

- Call the Display Information method to display information for all three employees using the following, for example,
  emp[0].DisplayInformation();

- Record the output of your program here:

- Call the Display Information method to display information for all three employees using a different approach, for example,
  emp0.DisplayInformation();

- Before you run your test program, write down your prediction of the output of your program here.
- Run your test program and record the output of your program here:

- Do you see any difference in the information displayed for the same employee (represented by emp[0] and emp0)? How about emp[1] and emp1, emp[2] and emp2? Explain WHY.