Lab 7: The Employee Class

Objectives

The objectives of this laboratory are

- to learn how to define a simple Java class with mutators and accessors,
- to learn how to develop a Java application involving conditional statements.

1. The class Employee

First, design and implement a class named Employee to represent employee information according to the following specification. As done in Laboratory 6, start from the design step.

Download and draw a UML diagram for the class and save it in PDF format as employee.pdf. Have your diagram checked by the instructor or a TA before you implement it.

Fields. Each Employee object has three fields: employee id(in int), employee name (in String), and a basic salary (in int).

Constructor. The constructor should have three parameters(id, name and basic salary).

Accessors. This class has an accessor for each of the three fields.

Mutators. This class has a mutator for each of the three fields.

Other methods. In addition to these methods, the employee class contains two more methods, namely, netSalary() and print(). The first method netSalary() calculates the net salary of the employee based on the basic salary and returns it. Hence, netSalary() will have a double return type. Net salary is calculated according to the following rules:

- House rent allowance(HRA) is 15% of the basic salary.
- Dearness allowance(DA) is 45% of the basic salary.
- Gross salary = basic salary + house rent allowance(HRA) + dearness allowance(DA)
- Income tax is 15% of the gross salary if the gross salary is less than $40,000 otherwise the income tax is 20% of the gross salary.
- Net salary = gross salary - income tax

The second method print() prints the employee information. For example, employee 2 with name “Brian Marvin” and basic salary $35,000, print() method will print the following information:

```
Here are the employee details...
Employee id :   2
Employee name :      Brian Marvin
Net salary :      $44800
```

2. Java application

Now, write an interactive Java application named EmployeeApplication that creates employees according to input given by the user. Your application should ask the user for the information of an employee and then instantiate an employee object. Use delimiter in the scanner to allow new line characters in the string input. For example,

```
Scanner sc = new Scanner(System.in).useDelimiter("\n");
```

Use the employee object to call the print method of employee class to print employee details: You application should behave the following way:

```
Enter employee id :   1
Enter employee name :      Dasa Marvin
Enter basic salary :      35000

Here are the employee details...
Employee id :   1
Employee name :      Dasa Marvin
Net salary :      $48000
```

Test your application with four different test cases. Save the output of your test runs in a text file named employee.txt.

3. Print in Java application

Now, suppose the employee information has been changed. In next part of your application, ask user for the updated employee information in the following way:

```
Enter updated employee id :   4
Enter updated employee name :      Dasa Marvin
Enter updated basic  salary :      55000
```

Since we need to update the existing employee information, use the previous employee object and it's mutators to update the employee information. Do not create any new employee object.

Now, suppose, we don't have the print method in the employee class. Write java statements in your application to print employee details using accessors and netSalary() method of employee class. For example, assuming the name of the employee object is emp, use accessor emp.getName() to print the employee name.

As you learned in Laboratory 6, both Java sources should be properly documented using Java comment blocks in the Javadoc format.

What to hand in

Upon completion of your laboratory, upload the following files:

- UML diagram: employee.pdf
- Java source: Employee.java, EmployeeApplication.java
- Output: employee.txt

Create a folder by your last name, copy your files inside the folder. Then create a zip file(your_last_name.zip) and upload the zip file in the moodle.