Salary System

Li Chengyao
5140219288

May 28, 2017

1 PROBLEM DESCRIPTION

This homework revisits the idea of superclass/subclass, polymorphism. And study how to use Arrays.sort(). **Remember you need to implement Comparable or Comparator to sort your object array.**
Write a superclass Worker and subclasses HourlyWorker and SalariedWorker. Every worker has a name, a salary rate, and the total salary he got so far from the beginning of his employment. Write a method computePay(int hours) that computes the weekly pay for every worker. An hourly worker get paid the the hourly wage for the actual number of hours worked, of hours is at most 40. If the hourly worker worked more than 40 hours, the excess is paid at double rate. The salaried worker gets paid the hourly wage for 40 hours, no matter what the actual number of hours is. The total salary is just the accumulated value of all the paid salary for the worker. Beside the computePay() method, you can add more methods if you like or think necessary.
After creating an array of Worker, I want to sort the array and print out the sorted result. By default, the workers are sorted by their name. You should also be able to sort it by their salary rate, or by the total salary they've got. This means if I use Arrays.sort(arrayname) directly, I'm sorting by name. Otherwise I sort by using different comparators.
When print out the information about the workers using System.out.println(worker), I wish to be able to see the worker's type too, i.e., whether he is HourlyWorker or SalariedWorker, and with his name, salary rate, and total salary.

2 ALGORITHM

2.1 WORKER

Similar to Project 2, there are three different classes about Workers in this program, Worker, HourlyWorker and SalariedWorker. Obviously HourlyWorker and SalariedWorker are extended from Worker.

- Worker

  This class helps to define basic variables like workerName, workTime, salaryRate and totalSalary. In the meantime methods such as printPay() and computePay() is basically implemented. When executing the constructor to input the information of the worker such as name and salary rate, several checks will be made to make sure that all the inputs are legal.

- HourlyWorker

  It is extended from Worker. Due to the work time threshold the computePay() method is overrided.

- SalaryWorker

  It is also extended from Worker. Because the effective work time is fixed for SalaryWorker the computePay() method is overrided as well.

2.2 Sort Methods

We use *Array.sort()* to sort by different properties. As is shown in problem description, the workers are sorted by their name by default. In the code we use *Array.sort(Workers)* directly to sort by name. It is implemented by implementing class Worker from class *Comparable*. In the *compareTo* method the string of name is compared by the order of the character. If we get a positive result we define that this one is larger than the compared one.
As for sorting by salary rate and total salary, we use class *Comparator* rather than *Comparable* to implement it. In the extended classes *salaryRateCompare* and *totalSalaryCompare* method *compare* is instantiated to create a new rule to compare. To sort by salary rate and total salary using code like *Array.sort(Workers, new salaryRateCompare())* is OK.

2.3 Main Method

We create 6 workers initially and define their characteristics by constructor directly. Then we use *Array.sort()* respectively to sort by different comparators and the display the result.

3 Results

3.1 Environment

- Windows 10
- Java Development Kit 1.8.0_131
- Eclipse

3.2 Screenshots of the Result

We use command line to compile and execute the program. The result is shown in Fig. 3.1-3.3.

3.3 Thoughts

This project helps us to learn the difference between *Comparable* and *Comparator*. In the meantime we try to implement them by ourselves, which contributes to our coding skills.

```
F:\OS Project\SalarySystem\src\com\salary>java SalarySystem
Sort by name:
Hourly worker : Ada
Salary rate: 21.0
Total salary: 1260.0

Salaried worker : Bryce
Salary rate: 30.0
Total salary: 1200.0

Hourly worker : Darwin
Salary rate: 20.0
Total salary: 600.0

Salaried worker : Iris
Salary rate: 31.0
Total salary: 1240.0

Salaried worker : TA
Salary rate: 25.0
Total salary: 1000.0

Hourly worker : Teacher
Salary rate: 40.0
Total salary: 960.0
```

Figure 3.1: Screenshot of Salary System (1)

```
Sort by salary rate:
Hourly worker : Darwin
Salary rate: 20.0
Total salary: 600.0

Hourly worker : Ada
Salary rate: 21.0
Total salary: 1260.0

Salaried worker : TA
Salary rate: 25.0
Total salary: 1000.0

Salaried worker : Bryce
Salary rate: 30.0
Total salary: 1200.0

Salaried worker : Iris
Salary rate: 31.0
Total salary: 1240.0

Hourly worker : Teacher
Salary rate: 40.0
Total salary: 960.0
```

Figure 3.2: Screenshot of Salary System (2)

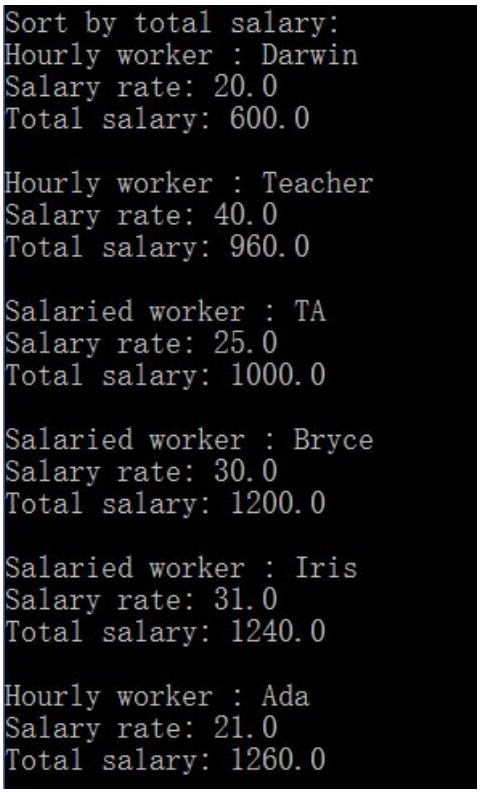

Figure 3.3: Screenshot of Salary System (3)