Assignment 2 Report

Berke Bayraktar
b21992847

April 2020

1 Problem Definition

I was expected to create a payroll system for a university in the *JAVA* programming language while also incorporating the *object oriented paradigm*.

2 Solution Approach

I used *inheritance* and *polymorphism* to solve the given problem. I have also created a well defined inheritance hierarchy with the help of *abstract* classes to better apply these OOP concepts to my program. This approach prevented me from writing duplicate code and made my program more extensible.

3 Project Structure

The university accommodates many different type of personnel which posses similar properties and behaviours therefore I created a inheritance hierarchy which contains every type of personnel in the university. I have also took advantage of *abstract* classes to define a more logical hierarchy.

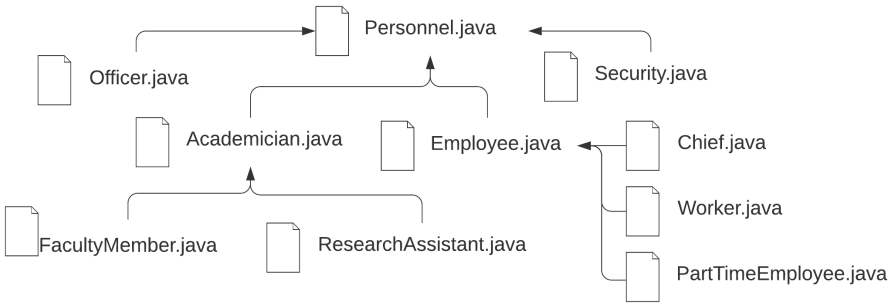

Figure 1: Overhead look of the inheritance hierarchy

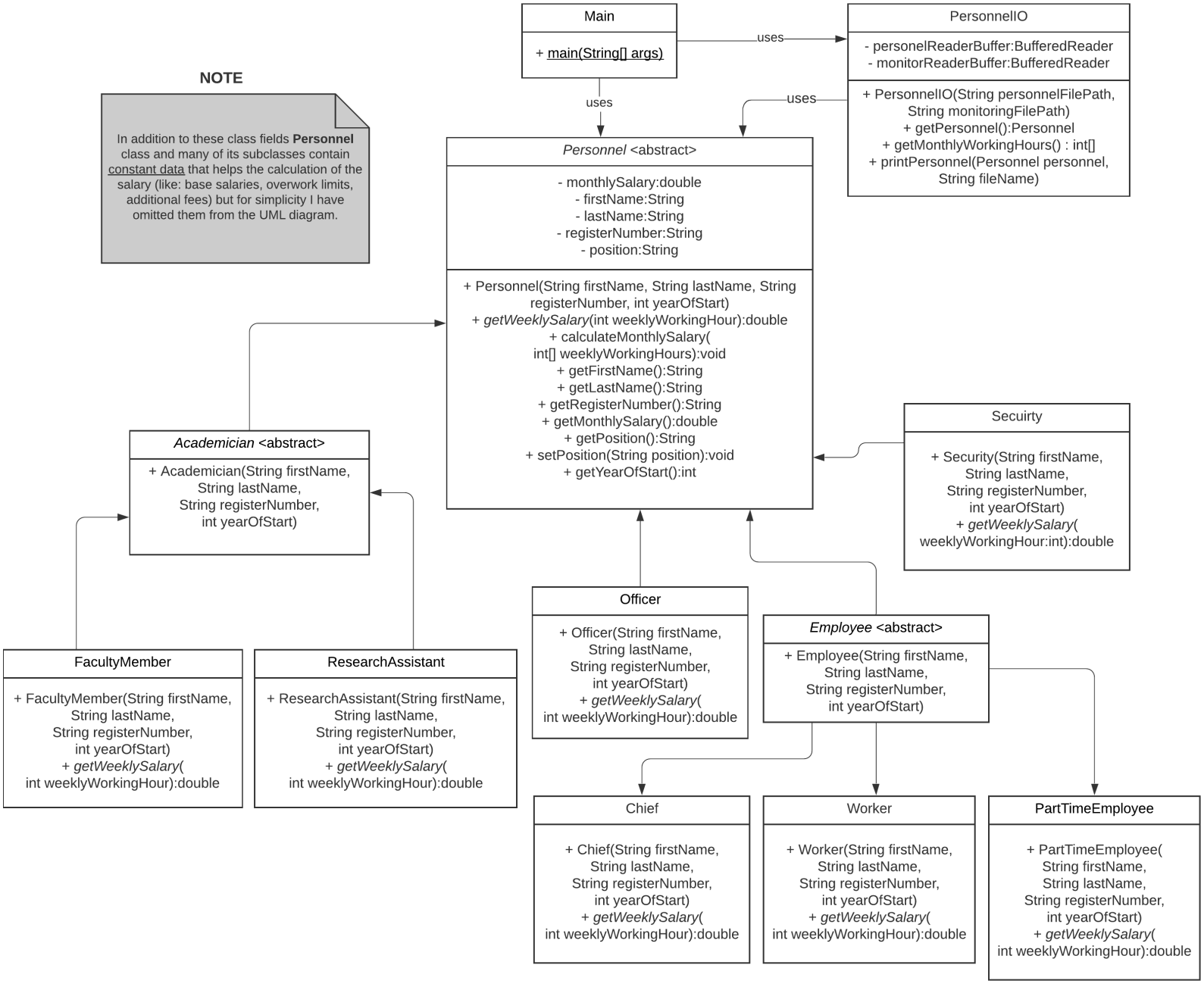

Figure 2: Project UML Diagram

4 Implementation

4.1 Reading the data

There are two input files: *personnel.txt* and *monitoring.txt*. **PersonnelIO** reads these files line by line. Each time a line is read from *personnel.txt* a **Personnel** object is created with upcasting.

```
Personnel personnel = new SomeTypeOfPersonnel(); //type depends on input
```

Later on this **Personnel** is accesed by Main class via **getPersonnel()** method of **PersonnelIO** class.

Similarly working hours for the corresponding personnel is read from *monitoring.txt* and this data is later accesed in the Main class via **getMonthlyWorkingHours()** method of **PersonnelIO** class.

4.2 Calculating the salary

The **Personnel** class posseses an *abstract* method to get what would be the *weekly* salary of that personnel.

```
//Personnel.java
public abstract double getWeeklySalary(int weeklyWorkingHours);
```

This method is overridden in every subclass of **Personnel** that is not abstract to implement the specific way of salary calculation for that type of personnel.

These methods are later on accessed in the **Personnel** class to calculate the *monthly* salary.

```
//Personnel.java
public void calculateMonthlySalary(int[] weeklyWorkingHours) {
        double monthlySalary = 0;

        for (int currentWeek : weeklyWorkingHours) {
            monthlySalary += this.getWeeklySalary(currentWeek);
        }
        //...other salary additions that are common in every personnel
        this.monthlySalary = monthlySalary;
    }
```

This method to calculate salary is accessed in the Main class for every single personnel and since they are polymorphic we can call the salary method directly.

```
//Main.java
Personnel personnel = personnelIO.getPersonnel();
int[] monthlyWorkingHours = personnelIO.getMonthlyWorkingHours();
personnel.calculateMonthlySalary(monthlyWorkingHours);
```