Updates to Academia and Corporate Representation Packages

ITI 1121 ASSIGNMENT 2

Mohamad Ahmad
300108630 | MAHMA060@UOTTAWA.CA

Hello Mr. Musk,

Thank you kindly for your feedback on the first iteration of the project. As you desired, I have added a Person class to the Employee package (see UML diagram below).

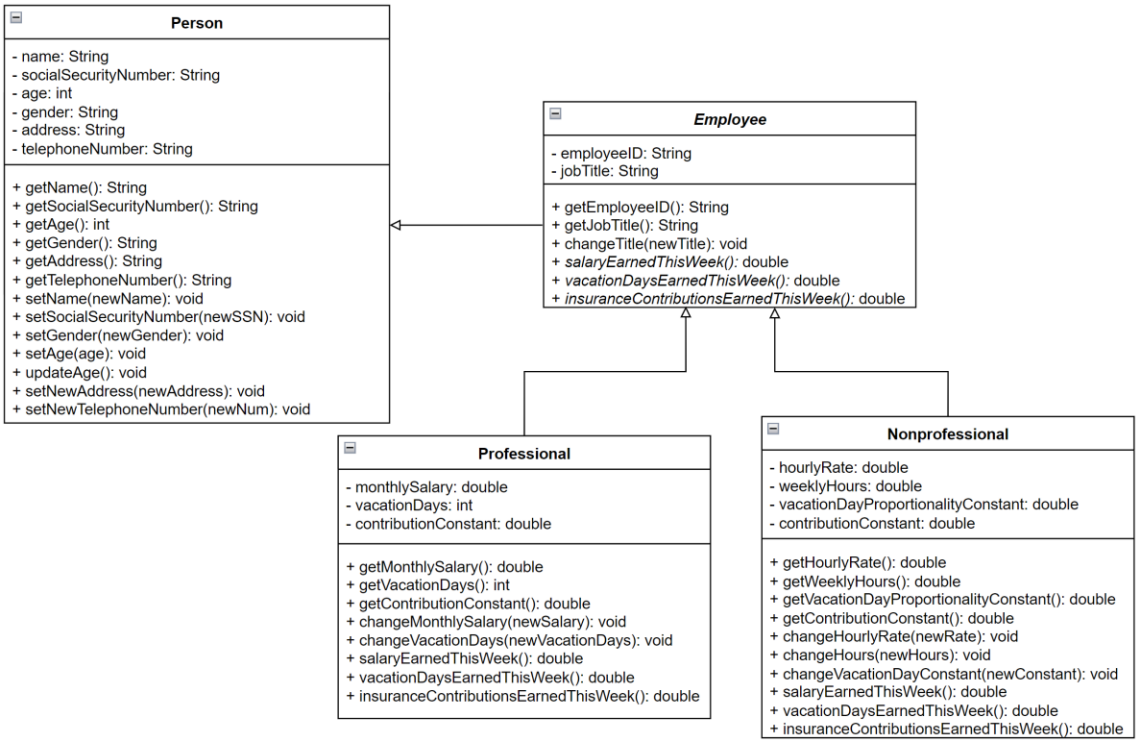

Figure 1: UML Diagram of Problem 2 (corporate enterprise employee package).

In aim to partition the components clearly and minimize coupling, all classes are in separate files (including the new person class). Furthermore, I have fulfilled your wishes to have unfilled positions accounted for. In other words, it is now possible to have a professional or non-professional position that is unfilled (i.e. not assigned to a person). These instances can be seen by information such as wages and vacation days being present, but things like name and social security number being NA. This change was achieved by using **4 cases of polymorphism**. Namely, each of the classes in the Employee package (person, employee, professional, and non-professional) have been endowed with **2 constructors** (Figure 2). The 2 options represent the case in which an instance of a staffed position is being made, or an instance of an unstaffed position is being made (in which all person-related instance variables are NA).

```
//constructor for filled position
public Employee(String name, String socialSecurityNumber, int age, String gender, String address, String telephoneNumber, String employeeID, String jobTitle){
    super(name, socialSecurityNumber, age, gender, address, telephoneNumber);
    this.employeeID = employeeID;
    this.jobTitle = jobTitle;
}

//constructor for unstaffed position
public Employee(String employeeID, String jobTitle){
    super();
    this.employeeID = employeeID;
    this.jobTitle = jobTitle;
}
```

Figure 2: Example of Polymorphism. Employee Class Constructors (lines 7 – 19).

Hence, when the desire for a new position is expressed, the job title, wages, etc. can be stored within the system **and only when we hire someone for the position will information like name and social security number be stored.**

As a consequence of this change, the instance variables name, social security number, age, and gender within the person class have now been endowed with setters. In aim to keep them as close to private variables as possible, these setters will only update the variables if they were prior NA.

```java
//setters with a one-time use (to be used when filling an unstaffed position)
public void setName(String newName){
    if (this.name == "UNSTAFFED"){
        this.name = newName;
    }
}
```

Figure 3: One-time use setters in the Person class (lines 58 – 62).

As for the first problem, minimal changes have been made to the functionality of the code (see the UML diagram below). However, some comments were added within the code for sake of better future maintainability and comprehension.

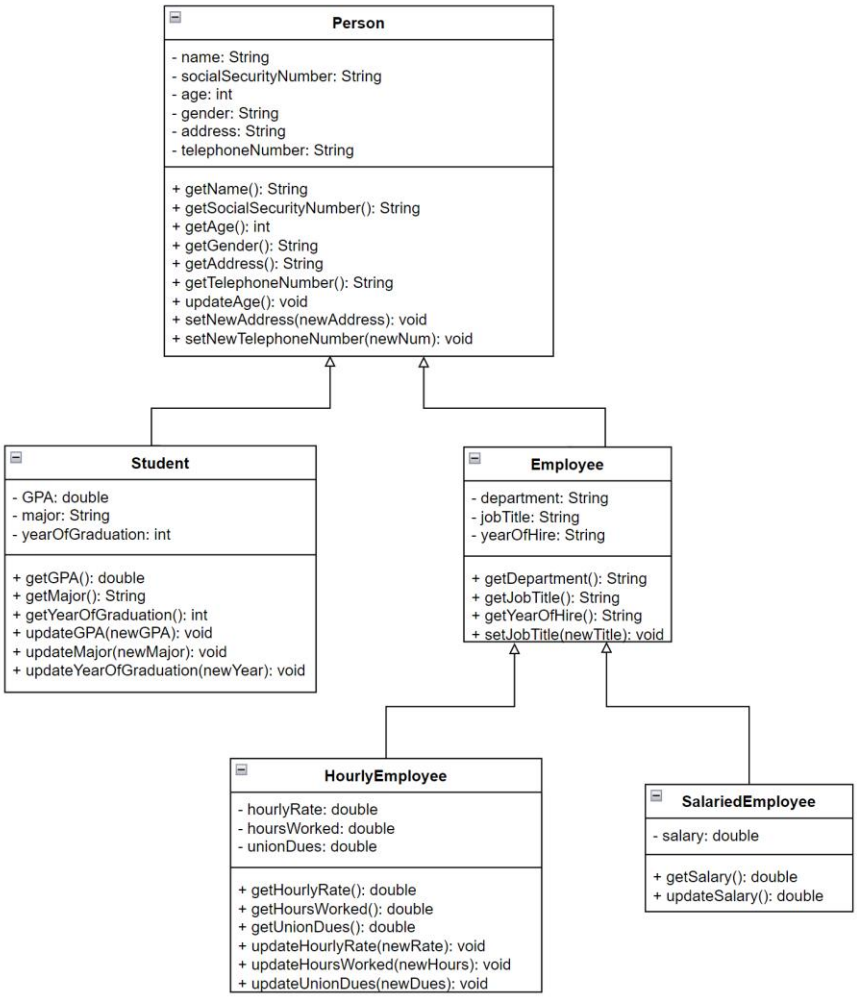

Figure 4: UML diagram of problem 1 (academia role package).

Overall, better OOP guidelines have been followed allowing future iterations of this project to be more seamless and less prone to error. The partitioning into separate files and clear commenting makes the package clear and free of too much complexity. The cases of polymorphism in problem 2 allow for a more robust corporate representation of the hiring process.