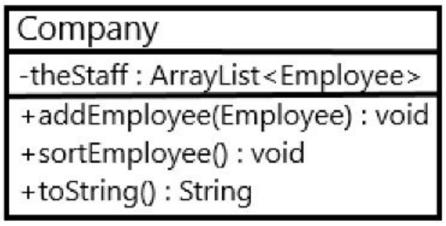
Lesson 5

Mark Eric Dieckmann

February 16, 2023

Lesson 5 revises abstract classes and interfaces. We develop a framework for a company (**Company**) to manage its employees. The abstract superclass **Employee** contains variables and methods that are shared by all employees. The company has three types of employees. Sales persons (**SalesPerson**) have a fixed number of hours. Service workers (**ServiceWorker**) and programmers (**Programmer**) have a fixed number of base hours and extra hours related to travel and overtime. The company can sort the employees alphabetically using Java's bubble sort. **Lesson5** is fully implemented.

1 Task: Implement Company


Company
-theStaff : ArrayList<Employee>
+addEmployee(Employee) : void
+sortEmployee() : void
+toString() : String

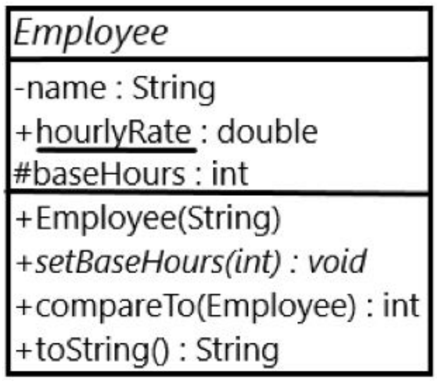
theStaff holds all employees.

addEmployee(arg) adds the Employee *arg* to the dynamic list.

sortEmployee() sorts the employees in *theStaff* alphabetically using bubble sort. Use for this purpose the *sort()* method of **Collections**.

toString() returns the string starting with "List:" followed by a line break followed by the return values of the *toString()* methods of all employees in *theStaff*.

2 Task: Implement Employee and SalesPerson



Abstract class names and methods are placed in italics.

name is the name of the employee. It is initialized with the constructor argument.

The class variable *hourlyRate* holds the standard salary for an hour of work.

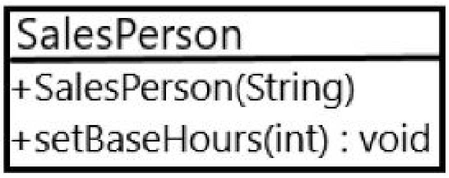
baseHours is the hours employees work on a fixed-time contract. All employees work for a fixed number of hours per week.

setBaseHours(arg) sets *baseHours*.

compareTo(arg) compares the names of two employees alphabetically. Implement for this purpose the interface **Comparable**.

toString() returns a string that lists all attributes of the employee. Sales persons work for a fixed number of hours and the return string should give for them "Name: " followed by *name* on 17 characters, followed by "Base salary: " followed by the product of *baseSalary* and *hourlyRate* on 6 float characters (2 after the comma). Workers with a mix of fixed hours and extra hours also have in their return string "Bonus: " followed by the value of the bonus on 6 float characters (2 after the comma). We discuss the computation of the bonus later.

Implement also the subclass **SalesPerson** that has only a fixed-hour contract.

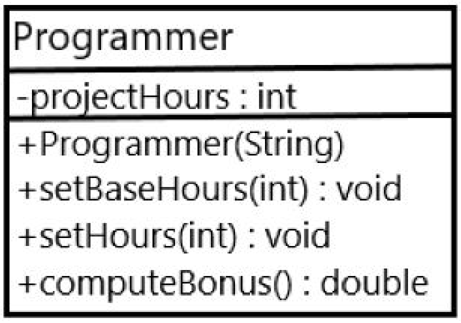


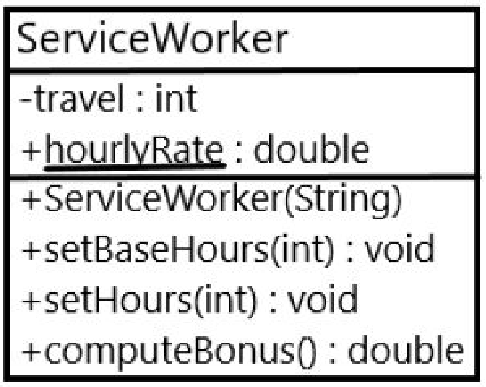
The constructor calls that of the superclass with its argument (the person's name).

setBaseHours(arg) sets the value of *baseHours*.

3 Task: Implement the classes Programmer and ServiceWorker

We want to enforce a standard on all subclasses of **Employee** that have a contract with a reduced number of base hours and take extra hours instead. For this purpose, we create our own interface **ContractWork** with the two abstract methods *public abstract double computeBonus()* and *public abstract void setHours(int)*. *public abstract* is the default in interfaces, so you can declare them as *double computeBonus()* and *void setHours(int)*.





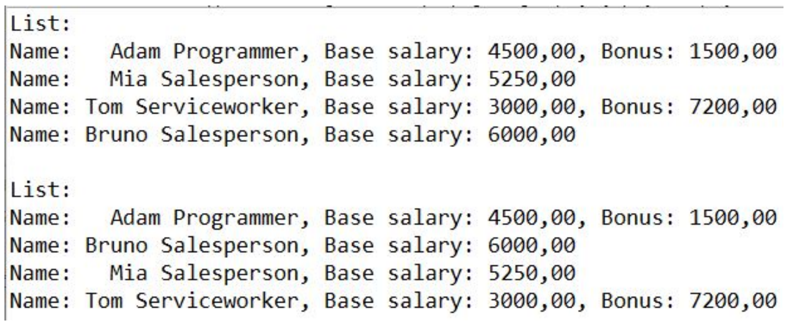
projectHours stores the extra hours for a programmer and *travel* that of the service worker. The programmer has the same salary per hour as the inherited *hourlyRate*. Service workers have an hourly rate that is 1.2 times that of *hourlyRate*. Use here a class variable that hides the inherited one.

The constructors of both classes call that of the superclass with their argument.

The inherited method *setHours(arg)* sets *projectHours* or *travel* to *arg*. The inherited *computeBonus()* returns the product of *projectHours* and the inherited *hourlyRate* in **Programmer**. That in **ServiceWorker** returns the product of *travel* and *hourlyRate*, which is defined in this class.

Use *computeBonus()* to compute the value after ”Bonus :” in the *toString()* method of **Employee**.

You should get a console output similar to that below.



```
List:
Name:   Adam Programmer, Base salary: 4500,00, Bonus: 1500,00
Name:   Mia Salesperson, Base salary: 5250,00
Name: Tom Serviceworker, Base salary: 3000,00, Bonus: 7200,00
Name: Bruno Salesperson, Base salary: 6000,00

List:
Name:   Adam Programmer, Base salary: 4500,00, Bonus: 1500,00
Name: Bruno Salesperson, Base salary: 6000,00
Name:   Mia Salesperson, Base salary: 5250,00
Name: Tom Serviceworker, Base salary: 3000,00, Bonus: 7200,00
```