# Lab: Encapsulation

Problems for exercises and homework for the ["Java OOP Basics" course @ SoftUni](https://softuni.bg/courses/java-oop-basics).

You can check your solutions here: <https://judge.softuni.bg/Contests/Practice/Index/475#0>.

## Sort Persons by Name and Age

Create a class **Person**, which should have **private** fields for:

* FirstName: String
* LastName: String
* Age: Integer
* ToString() - override

You should be able to use the class like this:

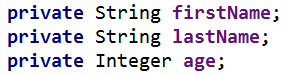
|  |
| --- |
| Main.java |
| **public static void** main(String[] args) **throws** IOException {  BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));  **int** n = Integer.*parseInt*(reader.readLine());   List<Person> persons = **new** ArrayList<>();   **for** (**int** i = 0; i < n; i++) {  String[] input = reader.readLine().split(**" "**);  persons.add(**new** Person(input[0], input[1], Integer.*parseInt*(input[2])));  }   Collections.*sort*(persons, (firstPerson, secondPerson) -> {  **int** sComp = firstPerson.getFirstName().compareTo(secondPerson.getFirstName());   **if** (sComp != 0) {  **return** sComp;  } **else** {  **return** firstPerson.getAge().compareTo(secondPerson.getAge());  }  });   **for** (Person person : persons) {  System.***out***.println(person.toString());  } } |

### Examples

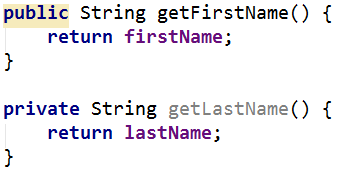
|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov 65  Boiko Borisov 57  Ventsislav Ivanov 27  Asen Harizanoov 44  Boiko Angelov 35 | Asen Harizanoov is a 44 yaers old.  Asen Ivanov is a 65 yaers old.  Boiko Angelov is a 35 yaers old.  Boiko Borisov is a 57 yaers old.  Ventsislav Ivanov is a 27 yaers old. |

### Solution

Create a **new class** and ensure **proper naming**. Define the **private** fields



Create getters and apply them access modifiers, which are as strictly as possible



Override **toString()** method:



## Salary Increase

Read person with their names, age and salary. Read percent bonus to every person salary. Persons younger than 30 get half bonus. Expand **Person** from previous task. Add **salary** **field** and **property** with proper **access**.

New **fields** and **methods**

* Salary: Double
* IncreaseSalary(Integer bonus)

You should be able to use the class like this:

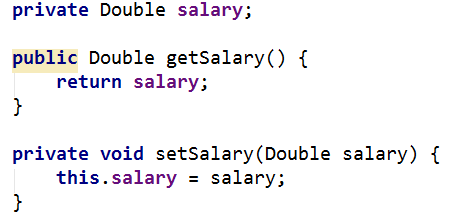
|  |
| --- |
| Main.java |
| **public static void** main(String[] args) **throws** IOException {  *//****TODO: Add reading logic***  **int** bonus = Integer.*parseInt*(reader.readLine());  **for** (Person person : persons) {  person.increaseSalary(bonus);  System.***out***.println(person.toString());  } } |

### Examples

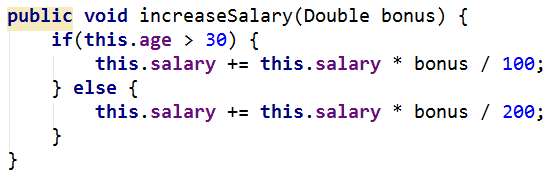
|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov 65 2200  Boiko Borisov 57 3333  Ventsislav Ivanov 27 600  Asen Harizanoov 44 666.66  Boiko Angelov 35 559.4  20 | Asen Ivanov get 2640.0 leva  Boiko Borisov get 3999.6 leva  Ventsislav Ivanov get 660.0 leva  Asen Harizanoov get 799.992 leva  Boiko Angelov get 671.28 leva |

### Solution

Add new **private** field for **salary** and proper **setters** and **getters** for it



Add new **method**, which will **update** salary with bonus



Refactor **toString()** method for this task.

## Validation Data

Expand Person with proper validation for every field:

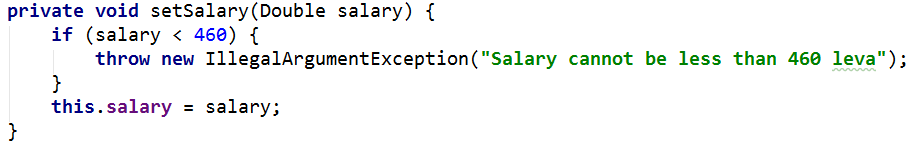
* **Names must be at least 3 symbols**
* **Age must not be zero or negative**
* **Salary can't be less than 460.0**

Print proper message to end user. Don't use System.out.println() in Person class.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov -6 2200  B Borisov 57 3333  Ventsislav Ivanov 27 600  Asen H 44 666.66  Boiko Angelov 35 300  20 | Age cannot be zero or negative integer  First name cannot be less than 3 symbols  Last name cannot be less than 3 symbols  Salary cannot be less than 460 leva  --------------------  Ventsislav Ivanov get 660.0 leva |

### Solution

Add validation to all setters in Person. Validation may look like this or something similar:

## First and Reserve Team

Create a Team class. Add to this team all person you read. All person younger than 40 go in first team, others go in reverse team. At the end print first and reserve team sizes.

The class should have **private fields** for:

* Name: String
* First Team Players: List<Person>
* Reserve Team Players: List<Person>

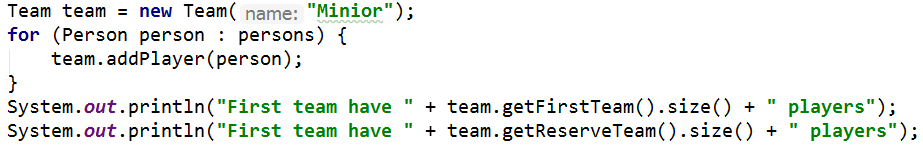
The class should have **constructors**:

* Person(String name)

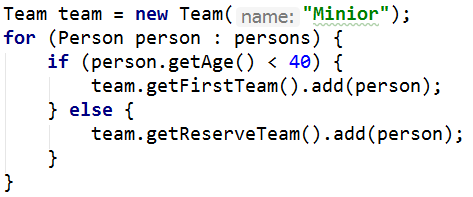
The class should also have **public methods** for:

* addPlayer(Person person): void
* getFirstTeam(): Collections.unmodifiableList
* getReserveTeam(): Collections.unmodifiableList

You should be able to use the class like this:



You should **NOT** be able to use the class like this:



### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov 20 2200  Boiko Borisov 57 3333  Ventsislav Ivanov 27 600  Grigor Dimitrov 25 666.66  Boiko Angelov 35 555 | First team have 4 players  Reserve team have 1 players |

### Solution

Add new class Team. Its fields and constructor look like:



Properties for FirstTeam and ReserveTeam have only getters:



There will be only one method, which add players to teams:

