# 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/475/Encapsulation-Lab> .

## Sort 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:

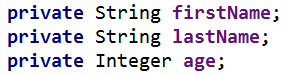
|  |
| --- |
| Main.java |
| **public static void** main(String[] args) **throws** IOException {  BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));  **int** n = Integer.*parseInt*(reader.readLine());   List<Person> people = **new** ArrayList<>();   **for** (**int**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

### Examples

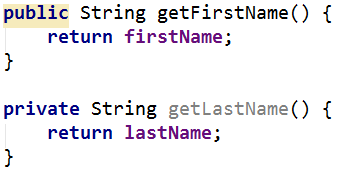
|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov 65  Boiko Borisov 57  Ventsislav Ivanov 27  Asen Harizanoov 44  Boiko Angelov 35 | Asen Harizanoov is 44 years old.  Asen Ivanov is 65 years old.  Boiko Angelov is 35 years old.  Boiko Borisov is 57 years old.  Ventsislav Ivanov is 27 years 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(Double bonus)

You should be able to use the class like this:

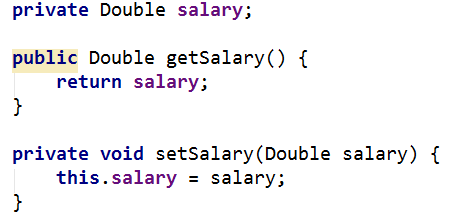
|  |
| --- |
| Main.java |
| **public static void** main(String[] args) **throws** IOException {  *//****TODO: Add reading logic***  **doube** bonus = Double.*parseDouble*(reader.readLine());  **for** (Person person : people) {  person.increaseSalary(bonus);  System.***out***.println(person.toString());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

### Examples

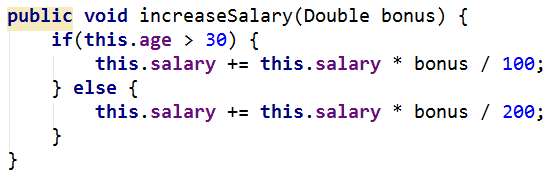
|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  Asen Ivanov 65 2200  Boiko Borisov 57 3333  Ventsislav Ivanov 27 600  Asen Harizanoov 44 666.66  Boiko Angelov 35 559.4  20 | Asen Ivanov get 2640.0 leva  Boiko Borisov get 3999.6 leva  Ventsislav Ivanov get 660.0 leva  Asen Harizanoov get 799.992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



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

**Note**: do not use **String.format()** in toString() method.

## 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**

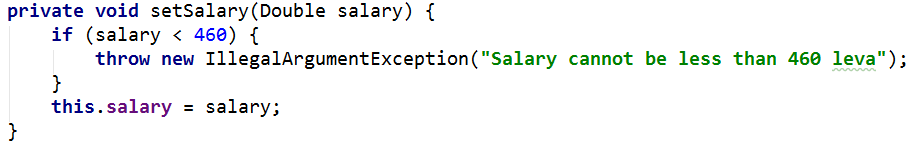
Print proper message to end user (look at example for messages).

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 gets 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>

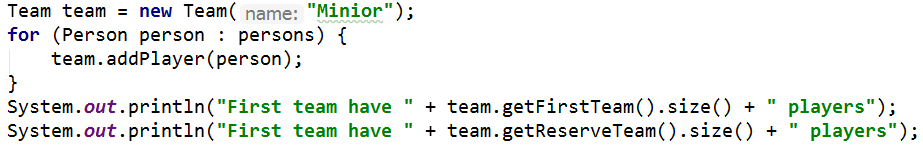
The class should have **constructors**:

* Team(String name)

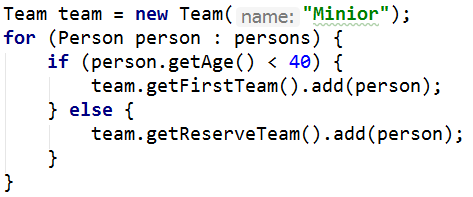
The class should also have **public methods** for:

* 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:

