# Lab: OOP

## Car Info

Create a class named Car.

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

* Brand: **text**
* Model: **text**
* Horsepower: **integer number**

**Create getters** and **setters** for each class field.

**Test the Program**

* The input consists of line - representing car object.
* You will have car info in the following format "**{brand} {model} {horsePower}**"separated by single space.
* Print car in the following format: **"The car is: {brand} {model} – {horsePower} HP."**

#### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Chevrolet Impala 390 | The car is: Chevrolet Impala - 390 HP. |
| Skoda Karoq 150 | The car is: This Skoda Karoq - 150 HP. |

## Opinion Poll

* Create a **Person** class with two fields **name** and **age**.
* Write a program that uses line of personal information in the following format "**{name} {age}"**
* Print the data in the following format: **"{name} is {age} years old."**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Peter 12 | Peter is 12 years old. |
| Sofia 33 | Sofia is 33 years old. |

## Bank Account

Create class BankAccount.

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

* Id: **int** (Starts from **1** and **increments** for every **new** **account**)
* Balance: **double**
* Interest rate: **double** (Shared for all accounts. **Default value: 0.02**)

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

* setInterestRate
* getInterest
* deposit

Create a test client supporting the following commands:

* **Create**
* **Deposit {Id} {Amount}**
* **SetInterest {Interest}**
* **GetInterest {ID} {Years}**
* **End**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| Create  Deposit 1 20  GetInterest 1 10  End | Account ID1 created  Deposited 20 to ID1  4.00 |  |
| Create  Create  Deposit 1 20  Deposit 3 20  Deposit 2 10  SetInterest 1.5  GetInterest 1 1  GetInterest 2 1  GetInterest 3 1  End | Account ID1 created  Account ID2 created  Deposited 20 to ID1  Account does not exist  Deposited 10 to ID2  30.00  15.00  Account does not exist | Sets the global interest rate to 1.  Prints interest for a bank account with id 1 for 1 year period. |

## Hierarchical Inheritance

Create three classes named Animal, Dog, and Cat.

* Animal with a single public method eat() that prints: **"eating…"**
* Dog with a single public method bark() that prints: **"barking…"**
* Cat with a single public method meow() that prints: **"meowing…"**
* Dog and Cat should inherit from Animal.

## Company Roster

Define a class **Employee** that holds the following information: **name, salary, position, department, email,** and **age**. The **name, salary**, **position,** and **department** are **mandatory,** while the rest are **optional**.

Your task is to write a program that takes **N** lines of information about employees from the console and calculates the department with the highest average salary, and prints for each employee in that department his **name, salary, email, and age** - **sorted by salary in descending order**. If an employee **doesn't have** an **email** – in place of that field, you should print "**n/a**" instead, if he doesn't have an **age** – print "**-1**" instead. Print in the following format:

**"Highest Average Salary: {department}**

**{name1} {salary1} {email1} {age1}**

**{name2} {salary2} {email2} {age2}**

**…**

**{namen} {salaryn} {emailn} {agen}"**

The **salary** should be printed to **two decimal places** after the separator.

**Hint**: You can define a **Department** class that holds a list of employees.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  Peter 120.00 Dev Development peter@abv.bg 28  Tina 333.33 Manager Marketing 33  Sam 840.20 ProjectLeader Development sam@sam.com  George 0.20 Freeloader Nowhere 18 | Highest Average Salary: Development  Sam 840.20 sam@sam.com -1  Peter 120.00 peter@abv.bg 28 |
| 6  Silver 496.37 Temp Coding silver@yahoo.com  Sam 610.13 Manager Sales  John 609.99 Manager Sales john@abv.bg 44  Venci 0.02 Director BeerDrinking beer@beer.br 23  Andre 700.00 Director Coding  Popeye 13.3333 Sailor SpinachGroup popeye@pop.ey | Highest Average Salary: Sales  Sam 610.13 n/a -1  John 609.99 john@abv.bg 44 |

## Pokemon Trainer

Define a class **Trainer** and a class **Pokemon**.

**Trainers** have:

* **Name**
* **Number of badges**
* **A collection of pokemon**

**Pokemon** have:

* **Name**
* **Element**
* **Health**

All values are **mandatory**. Every Trainer **starts with 0 badges**.

You will be receiving lines until you receive the command "**Tournament**". Each line will carry information about a pokemon and the trainer who caught it in the format:

**"{trainerName} {pokemonName} {pokemonElement} {pokemonHealth}"**

**TrainerName** is the name of the Trainer who caught the pokemon. Trainers' names are **unique**.  
After receiving the command "**Tournament**", you will start receiving commands until the "**End**" command is received. They can contain one of the following:

* **"Fire"**
* **"Water"**
* **"Electricity"**

For every command, you must check if a trainer has at least 1 pokemon with the given element. If he does, he receives 1 badge. Otherwise, all of his pokemon **lose 10 health**. If a pokemon falls **to 0 or less health**, **he dies** and must be deleted from the trainer's collection. In the end, you should print all of the trainers, **sorted by the number of badges they have in descending order** (if two trainers have the same amount of badges, they should be sorted by order of appearance in the input)in the format:

**"{trainerName} {badges} {numberOfPokemon}"**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Peter Charizard Fire 100  George Squirtle Water 38  Peter Pikachu Electricity 10  Tournament  Fire  Electricity  End | Peter 2 2  George 0 1 |
| Sam Blastoise Water 18  Narry Pikachu Electricity 22  John Kadabra Psychic 90  Tournament  Fire  Electricity  Fire  End | Narry 1 1  Sam 0 0  John 0 1 |