# Exercises: Interfaces and Abstraction

This document defines the exercises for ["Java OOP" course @ Software University](https://softuni.bg/modules/59/java-advanced). Please submit your solutions (source code) of all below described problems in [Judge](https://judge.softuni.bg/Contests/1582/Interfaces-and-Abstraction-Exercises).

## Define an Interface Person

Define an interface **Person** with methods getName and getAge. Define a class **Citizen** which implements **Person** and has a constructor which takes a String **name** and an int **age**.

Add the following code to your main method and submit it to Judge.

|  |  |
| --- | --- |
| **Citizen** | |
| - | name: String |
| - | age: int |
|  |  |
| + | Citizen (String, int) |
| + | getName() : String |
| + | getAge() : int |

|  |  |
| --- | --- |
| **<<Interface>>**  **Person** | |
| + | getName() : String |
| + | getAge() : int |

|  |
| --- |
| **public static void** main(String[] args) {  Class[] citizenInterfaces = Citizen.**class**.getInterfaces();  **if**(Arrays.*asList*(citizenInterfaces).contains(Person.**class**)){  Method[] fields = Person.**class**.getDeclaredMethods();  Scanner scanner = **new** Scanner(System.***in***);  String name = scanner.nextLine();  **int** age = Integer.*parseInt*(scanner.nextLine());  Person person = **new** Citizen(name,age);  System.***out***.println(fields.**length**);  System.***out***.println(person.getName());  System.***out***.println(person.getAge());  } } |

If you defined the interface and implemented it correctly, the test should pass.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Pesho  25 | 2  Pesho  25 |

## Multiple Implementation

Using the code from the previous task, define an interface Identifiablewith a **String** method getId and an interface Birthable with a **String** method getBirthDate and implement them in the Citizen class. Rewrite the **Citizen** constructor to accept the new parameters.

Add the following code to your main method and submit it to Judge.

|  |  |
| --- | --- |
| **<<Interface>>**  **Identifiable** | |
| + | getId(): String |

|  |  |
| --- | --- |
| **<<Interface>>**  **Birthable** | |
| + | getBirthDate(): String |

|  |  |
| --- | --- |
| **Citizen** | |
| - | name: String |
| - | age: int |
| - | id: String |
| - | birthDate: String |
| + | Citizen(String, int, String, String) |
| + | getName(): String |
| + | getAge(): int |
| + | getId(): String |
| + | getBirthDate(): String |
| + | toString(): String |

|  |  |
| --- | --- |
| **<<Interface>>**  **Person** | |
| + | getName(): String |
| + | getAge(): int |

|  |
| --- |
| **public static void** main(String[] args) {  Class[] citizenInterfaces = Citizen.**class**.getInterfaces();  **if** (Arrays.*asList*(citizenInterfaces).contains(Birthable.**class**)  && Arrays.*asList*(citizenInterfaces).contains(Identifiable.**class**)) {  Method[] methods = Birthable.**class**.getDeclaredMethods();  methods = Identifiable.**class**.getDeclaredMethods();  Scanner scanner = **new** Scanner(System.***in***);  String name = scanner.nextLine();  **int** age = Integer.*parseInt*(scanner.nextLine());  String id = scanner.nextLine();  String birthDate = scanner.nextLine();  Identifiable identifiable = **new** Citizen(name,age,id,birthDate);  Birthable birthable = **new** Citizen(name,age,id,birthDate);  System.***out***.println(methods.**length**);  System.***out***.println(methods[0].getReturnType().getSimpleName());  System.***out***.println(methods.**length**);  System.***out***.println(methods[0].getReturnType().getSimpleName());  } } |

If you defined the interfaces and implemented them, the test should pass.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Pesho  25  9105152287  15/05/1991 | 1  String  1  String |

## Birthday Celebrations

It is a well known fact that people celebrate birthdays, it is also known that some people also celebrate their pets birthdays. Extend the program from your last task to add **birthdates** to citizens and include a class **Pet**, pets have a **name** and a **birthdate**. Also create class Robot which has an **id** and **model**. Encompass repeated functionality into interfaces and implement them in your classes.

You will receive from the console an unknown amount of lines until the command "**End**" is received, each line will contain information in one of the following formats **"Citizen <name> <age> <id> <birthdate>"** for citizens,   
"**Robot** **<model> <id>**" for robots or "**Pet <name> <birthdate>**" for pets. After the end command on the next line you will receive a single number representing **a specific year**, your task is to print all birthdates (of both citizens and pets) in that year in the format **day/month/year** (the order of printing doesn’t matter).

|  |  |
| --- | --- |
| **<<Interface>>**  **Birthable** | |
| + | getBirthDate(): String |

|  |  |
| --- | --- |
| **<<Interface>>**  **Identifiable** | |
| + | getId(): String |

|  |  |
| --- | --- |
| **Citizen** | |
| - | name: String |
| - | age: int |
| - | id: String |
| - | birthDate: String |
| + | Citizen(Stirng, int, String, String) |
| + | getName(): String |
| + | getAge(): int |
| + | getId(): String |

|  |  |
| --- | --- |
| **Robot** | |
| - | id: String |
| - | model: String |
| + | Robot(Stirng, String) |
| + | getId(): String |
| + | getModel(): String |

|  |  |
| --- | --- |
| **Pet** | |
| - | name: String |
| - | birthDate: String |
| + | Pet(Stirng, String) |
| + | getName(): String |
| + | getBirthDate(): String |

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Citizen Pesho 22 9010101122 10/10/1990  Pet Sharo 13/11/2005  Robot MK-13 558833251  End  1990 | 10/10/1990 |
| Citizen Stamat 16 0041018380 01/01/2000  Robot MK-10 12345678  Robot PP-09 00000001  Pet Topcho 24/12/2000  Pet Kosmat 12/06/2002  End  2000 | 01/01/2000  24/12/2000 |
| Robot VV-XYZ 11213141  Citizen Penka 35 7903210713 21/03/1979  Citizen Kane 40 7409073566 07/09/1974  End  1975 | <no output> |

## Telephony

You have a business - **manufacturing cell phones**. But you have no software developers, so you call your friends and ask them to help you create a cell phone software. They agree and you start working on the project. The project consists of one main **model - a Smartphone**. Each of your smartphones should have functionalities of **calling other phones** and **browsing in the world wide web.**

Your friends are very busy, so you decide to write the code on your own. Here is the mandatory assignment:

You should have a **model** - **Smartphone** and two separate functionalities which your smartphone has - to **call other phones** and to **browse in the world wide web**. You should end up with **one class** and **two interfaces**.

|  |  |
| --- | --- |
| **<<Interface>>**  **Callable** | |
| + | call(): String |

|  |  |
| --- | --- |
| **<<Interface>>**  **Browsable** | |
| + | browse(): String |

|  |  |
| --- | --- |
| **Smartphone** | |
| - | numbers: List<String> |
| - | urls: List<String> |
| + | Smartphone(List<String>, List<String>) |
| + | call(): String |
| + | browse(): String |

### Input

The input comes from the console. It will hold two lines:

* **First line**: **phone numbers** to call (String), separated by spaces.
* **Second line: sites** to visit (String), separated by spaces.

### Output

* First **call all numbers** in the order of input then **browse all sites** in order of input
* The functionality of calling phones is printing on the console the number which are   
  being called in the format: **"Calling... <number>"**
* The functionality of the browser should print on the console the site in format:

**"Browsing: <site>!"** (pay attention to the exclamation mark when printing URLs)

* If there is a number in the input of the URLs, print: **"****Invalid URL!"** and continue printing the rest of the URLs.
* If there is a character different from a digit in a number, print: **"Invalid number!"** and continue to the next number.

### Constraints

* Each site's URL should consist only of letters and symbols (**No digits are allowed** in the URL address)

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 0882134215 0882134333 08992134215 0558123 3333 1  http://softuni.bg http://youtube.com http://www.g00gle.com | Calling... 0882134215  Calling... 0882134333  Calling... 08992134215  Calling... 0558123  Calling... 3333  Calling... 1  Browsing: http://softuni.bg!  Browsing: http://youtube.com!  Invalid URL! |