# Lab: Classes

Problems for in-class lab for the "JavaScript Advanced" course @ SoftUni. Submit your solutions in the SoftUni judge system at https://judge.softuni.bg/Contests/2768/Classes-Lab.

## 1. Person

Write a **class** that represents a personal record. It has the following properties, all set from the constructor:

- `firstName`
- `lastName`
- `age`
- `email`

And a method `toString()`, which prints a summary of the information. See the example for formatting details.

### Input

The constructor function will receive valid parameters.

### Output

The `toString()` method should **return** a string in the following format:

`` `{firstName} {lastName} (age: {age}, email: {email})` ``

Submit the class definition as is, **without** wrapping it in any function.

### Example

| Sample Input |
| --- |
| ```let person = new Person('Anna', 'Simpson', 22, 'anna@yahoo.com');
console.log(person.toString());``` |
| **Output** |
| `Anna Simpson (age: 22, email: anna@yahoo.com)` |

## 2. Get Persons

Write a function that returns an array of **Person** objects. Use the class from the previous task, create the following instances, and return them in an array:

| First Name | Last Name | Age | Email |
| --- | --- | --- | --- |
| **Anna** | Simpson | 22 | anna@yahoo.com |
| **SoftUni** | | | |
| **Stephan** | Johnson | 25 | |
| **Gabriel** | Peterson | 24 | g.p@gmail.com |

For any empty cells, do not supply a parameter (call the constructor with fewer parameters).

## Input / Output

There will be **no input**, the data is static and matches the table above. As **output**, **return an array** with **Person instances**.

Submit a function that returns the required output.

# 3. Circle

Write a **class** that represents a `Circle`. It has only one data property - its `radius`, and it is set through the **constructor**. The class needs to have **getter** and **setter** methods for its `diameter` - the setter needs to calculate the radius and change it and the getter needs to use the radius to calculate the diameter and return it.

The circle also has a getter **area()**, which calculates and **returns** its area.

## Input

The constructor function and diameter setter will receive valid parameters.

## Output

The **diameter()** and **area()** getters should **return** numbers.

Submit the class definition as is, **without** wrapping it in any function.

## Examples

| Sample Input | Output |
|---|---|
| ```let c = new Circle(2);```<br>```console.log(`Radius: ${c.radius}`);```<br>```console.log(`Diameter: ${c.diameter}`);```<br>```console.log(`Area: ${c.area}`);```<br>```c.diameter = 1.6;```<br>```console.log(`Radius: ${c.radius}`);```<br>```console.log(`Diameter: ${c.diameter}`);```<br>```console.log(`Area: ${c.area}`);``` | ```Radius: 2```<br>```Diameter: 4```<br>```Area: 12.566370614359172```<br>```Radius: 0.8```<br>```Diameter: 1.6```<br>```Area: 2.0106192982974678``` |

# 4. Point Distance

Write a JS **class** that represents a **Point**. It has **x** and **y** coordinates as properties, that are set through the constructor, and a **static method** for finding the distance between two points, called **distance()**.

## Input

The **distance()** method should receive two **Point** objects as parameters.

## Output

The **distance()** method should **return** a number, the distance between the two-point parameters.

Submit the class definition as is, **without** wrapping it in any function.

## Example

| Sample Input | Output |
|---|---|
| ```let p1 = new Point(5, 5);```<br>```let p2 = new Point(9, 8);```<br>```console.log(Point.distance(p1, p2));``` | 5 |