Exercises: Class Inheritance and Prototypes

Problems for exercises and homework for the "JavaScript Advanced" course @ SoftUni. Submit your solutions in the SoftUni judge system at https://judge.softuni.bg/Contests/339/.

1. Person and Teacher

Write a JS class **Person** and a class **Teacher** which extends **Person**. A **Person** should have a **name** and an **email**. A Teacher should have a **name**, an **email**, and a **subject**.

Input

There will be no input.

Output

Your function should return an object containing the classes **Person** and **Teacher**.

Example

template.js

```
function personAndTeacher() {
    //TODO

    return {
        Person,
        Teacher
    }
}
```

2. Inheriting and Replacing ToString

Extend the **Person** and **Teacher** from the previous task and add a class **Student** inheriting from **Person**. Add **toString()** functions to all classes, the formats should be as follows:

- Person - returns **"*Person (name: {name}, email: {email})*"**
- Student - returns ***"Student (name: {name}, email: {email}, course: {course})"***
- Teacher - returns ***"Teacher (name: {name}, email:{email}, subject:{subject})"***

Try to reuse code by using the **toString** function of the base class.

Input

There will be no input.

Output

Your function should return an object containing the classes **Person**, **Teacher** and **Student**.

Example

template.js

```
function toStringExtension() {
    //TODO
```

```
    return {
        Person,
        Teacher,
        Student
    }
}
```

3. Extend Prototype

Write a JS function which receives a **class** and attaches to it a property **species** and a function **toSpeciesString()**. When called, the function returns a string with format:

I am a <species>. <toString()>

The function **toString** is called from the current instance (call using **this**).

Input

Your function will receive a class whose prototype it should extend.

Output

There is no output, your function should only attach the properties to the given class' prototype.

Example

template.js

```
function extendClass(classToExtend) {
    //TODO
}
```

4. Class Hierarchy

Write a JS function that returns 3 classes - **Figure**, **Circle**, **Rectangle**.

Figure:

- should be abstract (cannot be instantiated)

Circle:

- extends Figure.
- has a property **radius**
- overrides **area** getter to return the area of the Circle (PI * r * r)
- **toString()** - should return a string representation of the figure in the format **"{type} - radius: {radius}"**

Rectangle

- extends Figure
- has properties **width** and **height**
- overrides **area** getter to return the area of the Rectangle (width * height)
- **toString()** - should return a string representation of the figure in the format **"{type} - width: {width}, height: {height}"**

Input

There will be no input.

Output

Your function should return an object containing the **`Figure`**, **`Circle`** and **`Rectangle`** classes.

Examples

This code demonstrates how your classes should behave:

Sample Code

```
let f = new Figure();        //Error
let c = new Circle(5);
console.log(c.area);         //78.53981633974483
console.log(c.toString());   //Circle - radius: 5
let r = new Rectangle(3,4);
console.log(r.area);         //12
console.log(r.toString());   //Rectangle - width: 3, height: 4
```