# Lab: Functions and Forms

Problems for in-class lab for the [“Technology Fundamentals” course @ SoftUni](https://softuni.bg/trainings/2056/technology-fundamental-september-2018#lesson-9622).

Submit your solutions in the SoftUni judge system at: [Functions-and-Forms-Lab](https://judge.softuni.bg/Contests/1230/Functions-and-Forms-Lab)

# Declaring and Invoking Functions

## Car tax calculator

Write a function that **receives a power** in kWof car’s**,** and **prints** the tax you have to pay in lv.

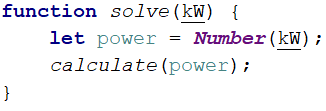
* under 37 kW - 0.43 lv./kW
* 37.01 – 55 kW - 0.50 lv./kW
* 55.01 – 74.00 - 0.68 lv./kW
* 74.01 – 110.00 - 1.38 lv./kW
* up 110.00 - 1.54 lv./kW

### Examples

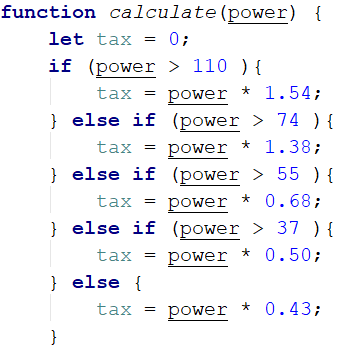
|  |  |
| --- | --- |
| **Input** | **Output** |
| 57.50 | 39.10 lv. |
| 90 | 124.20 lv. |
| 255.9 | 394.09 lv. |

### Hint

Receive the power in kW from the console and pass it to a function



Then create the method and make the if statements for each case



## Car tax calculator II

Add a new tax calculation function to the previous code, according to the age of the vehicle. Second parameter is a **car age**. Multiply by tax on the previous task. The coefficient depends on the age of the vehicle as follows. Print the result.

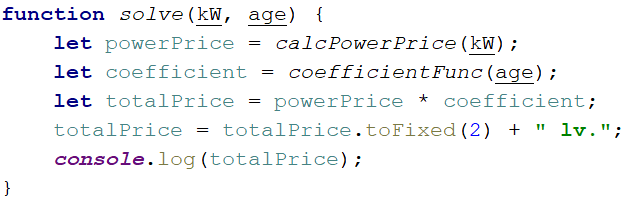
* under 5 year - 2.80
* 5 – 14 year - 1.50
* up to– 14 - 1.00

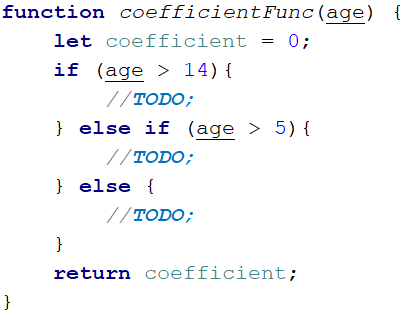
### Example

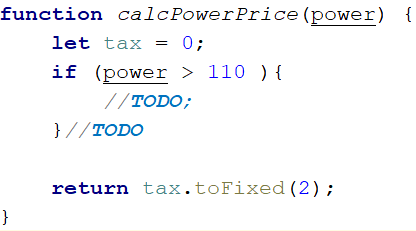
|  |  |
| --- | --- |
| **Input** | **Output** |
| 45  10 | 33.75 lv. |
| 90  7 | 186.30 lv. |
| 310  0.5 | 1336.72 lv. |

### Hints

Receive two parameters, and then make an additional function to calculator.







## Car tax calculator III

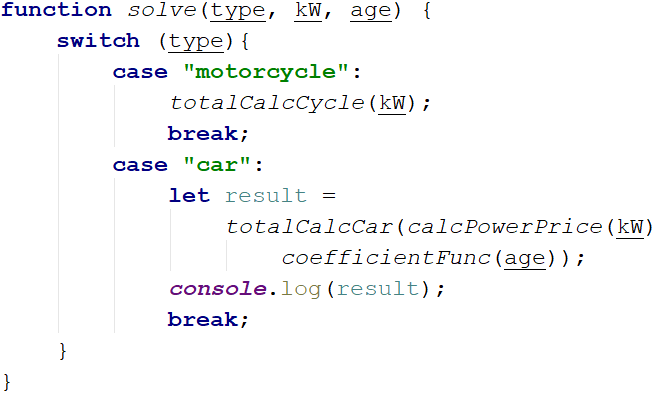
Add a new functionality to the calculator for the **motorcycle tax**. First parameter is **type** of vehicle, second is for **volume** (for motorcycle’s engine) or **power** (for the car’s engine). Third is for **years** like second task.

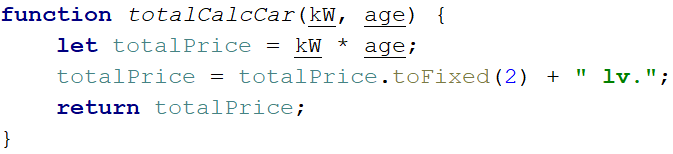
### Examples

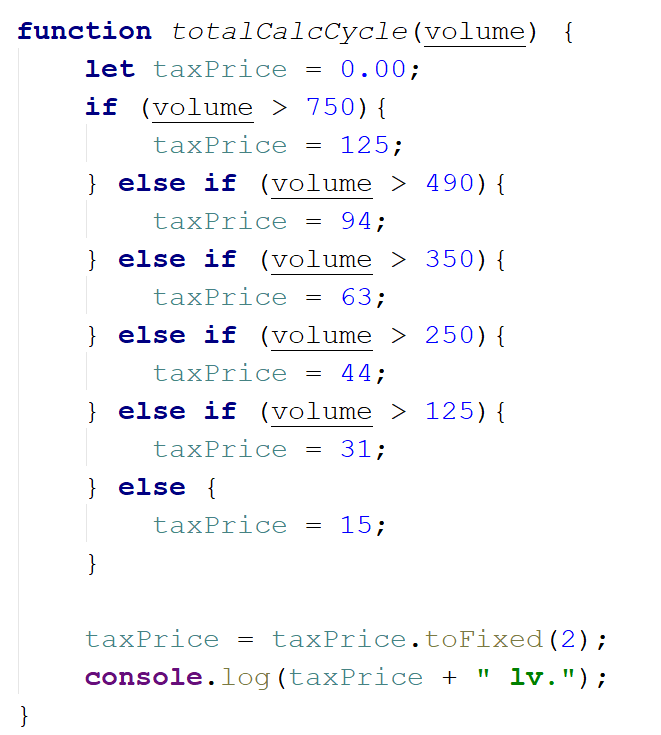
|  |  |
| --- | --- |
| **Input** | **Output** |
| 'motorcycle'  450  10 | 63.00 lv. |
| 'car'  90  7 | 186.30 lv. |

### Hints

1. You can use a function with parameters of functions (nested functions).







Write the rest of the functions

## Simple calculator

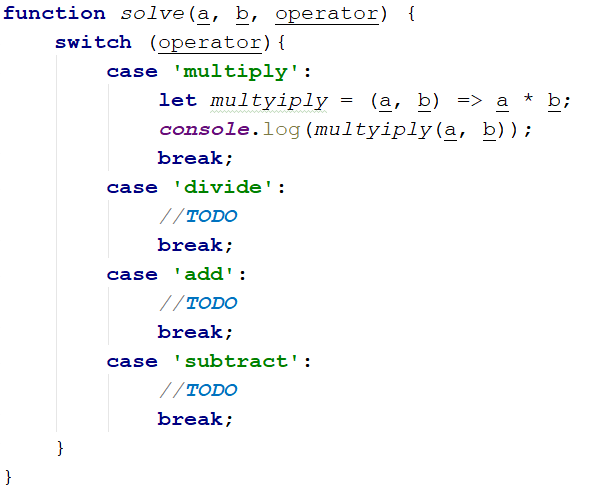
Write a function that **receives a three parameters** and write an arrows function that calculate result depends of operator. Operator can be ‘multiply’, ‘divide’, ‘add’, ‘subtract’.

The input comes as parameters named **numOne, numTwo, operator**.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  5  'multiply' | 25 |
| 40  8  'divide' | 5 |
| 12  19  'add' | 31 |
| 50  13  'subtract' | 37 |

### Hints

Use switch statements for the different operators



## Wrong result

You are given a JS function, that calculate the result of **numOne \* numTwo \* numThree** (the product) is **negative** or **positive**. Try to do this **WITHOUT** multiplying the 3 numbers.

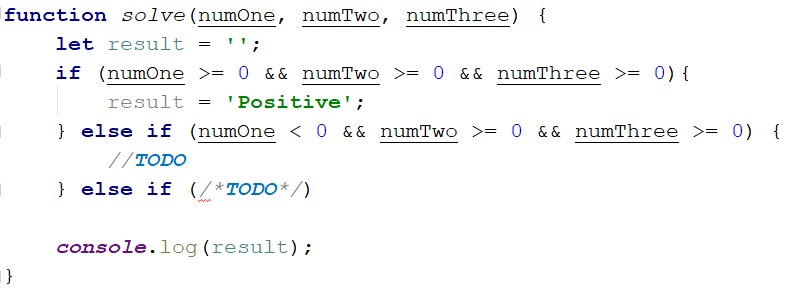
The input comes as parameters named **numOne \* numTwo \* numThree**.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  12  -15 | Negative |
| -6  -12  14 | Positive |
| -1  -2  -3 | Negative |
| -1  0  1 | Positive |

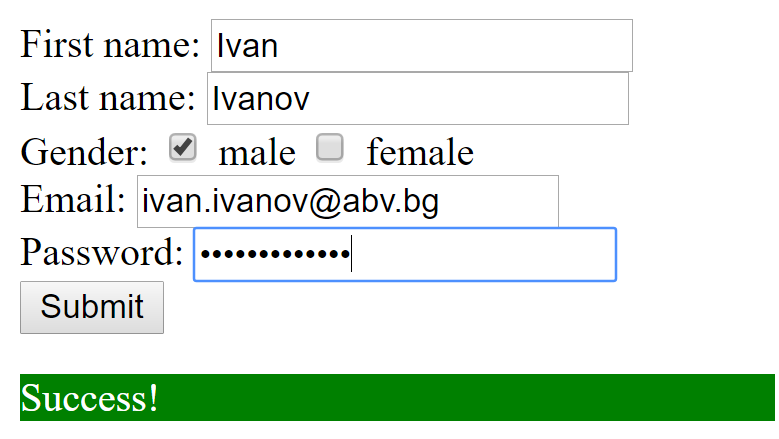
### Hints

Check all the different variantions for the three numbers

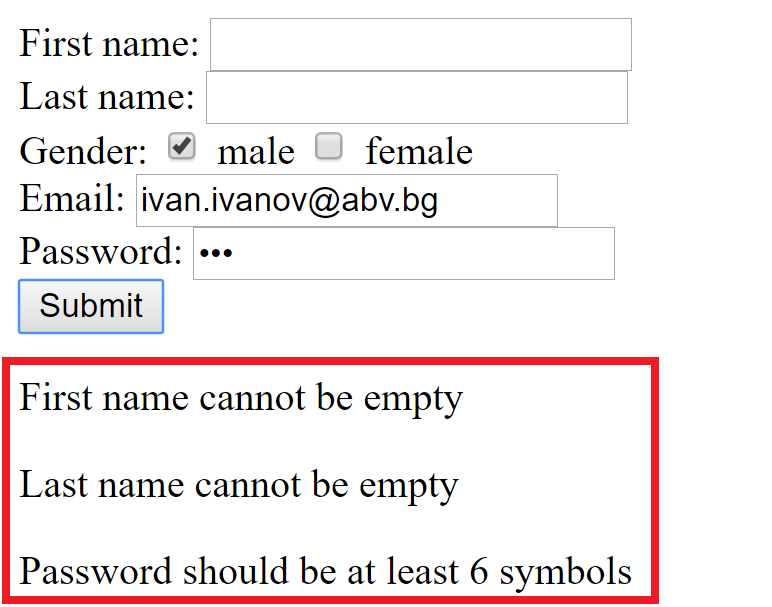


# Forms

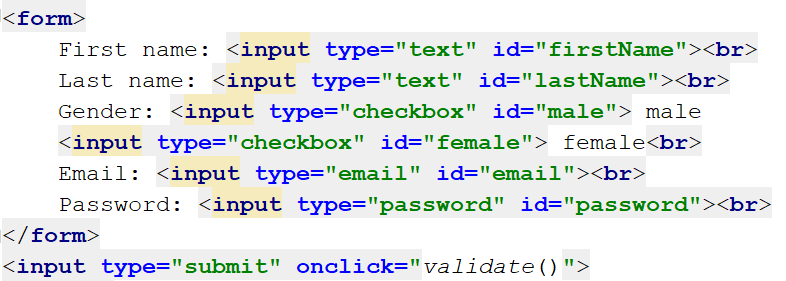
## Creating a basic form

We are going to create a form that takes some personal data about someone. At the end it should look like this:

If there are **invalid** inputs we will print messages like this:



## Creating the form

First let us create the HTML form. Create an **empty html file** in WebStorm and write the following **form in the body**:

* The **first** and **last** names will be **texts**
* For the **gender** we will create two **check boxes**
* The **email** will be input of **type email**
* The **password** will be input **type password**
* The **submit** input will be **outside of the form**, so the page does **not reload when it is clicked**
* When it **is clicked** we want to **execute a function.** We will write it in a bit.
* Each **input** will **have an id**, so we can access it with **DOM** when needed.

If you now run the html file you will see the form, but when you click on the button, **nothing will happen**. So the next step is to **write the validate function**.

## Getting all the form values

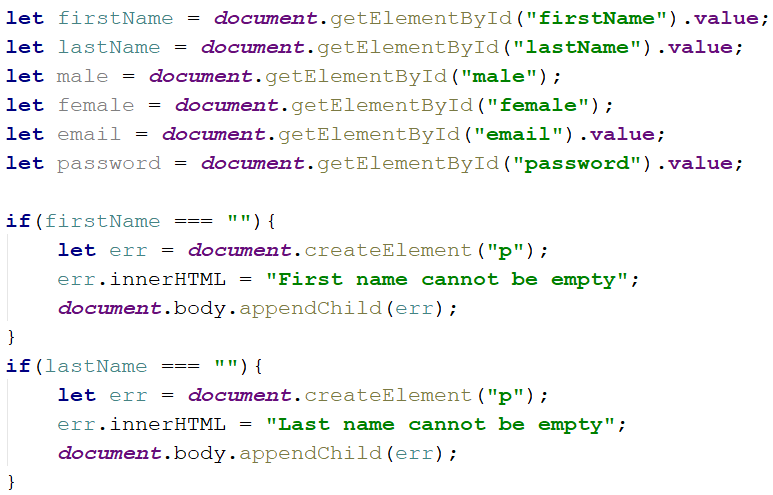
Now let us open a **script tag** right below the submit input and get all the values from the form and put them in variables:



* Since we have id-s for all the inputs we use **DOM to get their values**

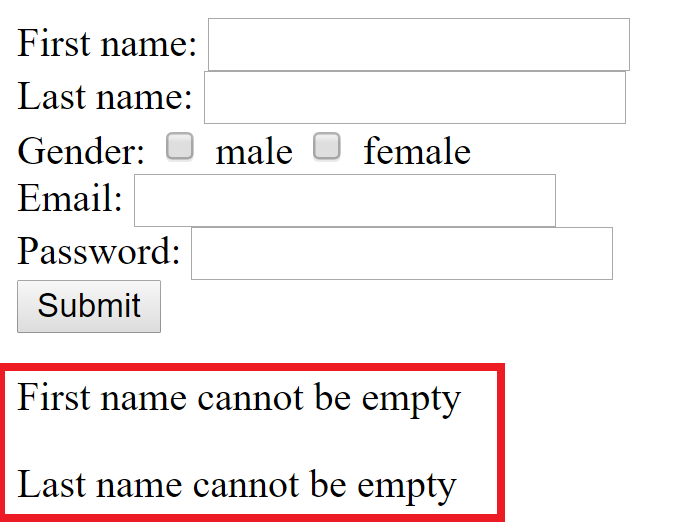
## Validating the names

Let us add **two if statements** to check if the **values are empty** and attach a **paragraph** with error **message** to the body if they are



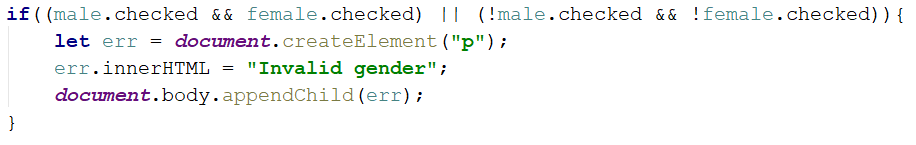
* We create a new element using the **createElement** function and pass it the tag of the element. In our case **paragraph (p)**
* We set the **innerHTML** (the **text content**) of the error to a corresponding message
* We **append** the **error paragraph** to the **body of the HTML**

If we test the code now and leave the two input fields empty we will see the errors in the browser:



## Validating the checkboxes

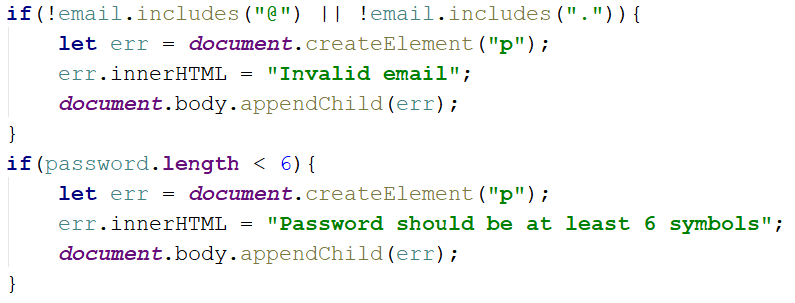
Now we will check if the person has chosen only **one of the gender options** (should have at least one and cannot have both).



* As we did with the names, we **create a paragraph**; we create an appropriate message and attach the whole element to the body of the HTML

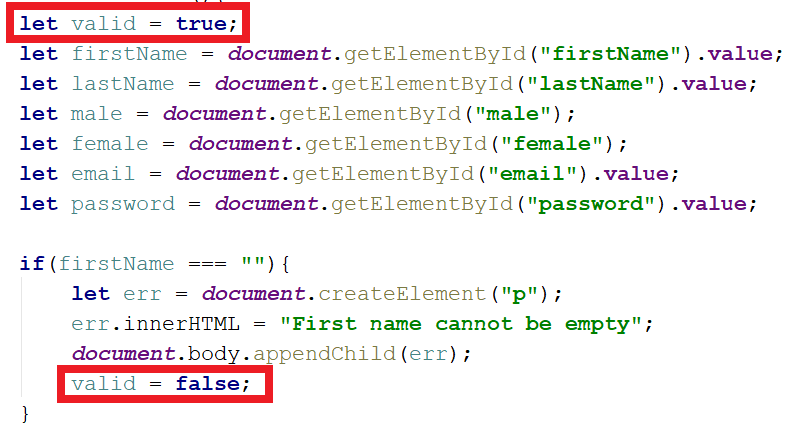
## Validating email and password

Finally let us validate the email and the password. We consider an **email for valid** if it contains at least **one "@"** and **one "."** For a **password to be valid**, we will want it to have **at least 6 symbols**.

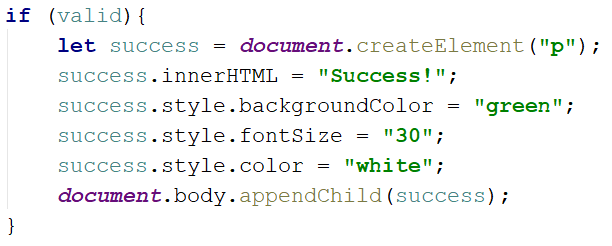


## The message for success

If all the input fields are correctly filled in, we will add a **success message**. To do that however we need to know if **all the fields are correct**. So, let us create a **Boolean variable** which will check whether **all the inputs are correct**. Create the variable at **the beginning of the function** and set it to **true**. In **each if statement** add a line that changes the value of the **Boolean variable to false**. (if something is invalid the value of the variable will be changed and we will know that we received invalid data)



* **Add the line to the rest of the if statements**



* If at the end of the function the **"valid"** variable is **still true**, we **append** a **success message**
* It will have content **"Success!"**
* The **background color** will be **green**
* The **font size** will be **30**
* The **color of the text** will be **white** (more visible)

***You have successfully created your first form! You can test it out***

