# Lab: Functions and Logic Flow

Problems for in-class lab for the ["JavaScript Fundamentals" course @ SoftUni](https://softuni.bg/trainings/2080/js-fundamentals-september-2018). Submit your solutions in the SoftUni judge system at [***//***](https://judge.softuni.bg/Contests/315/) ***TODO***.

## Multiplication Table

Write a JS function that takes **two integers** as an input.

* **The first parameter** will be the **starting number** that needs to be **multiplied**.
* **The second parameter** will be the **multiplier**.

Your task is to create a **multiplication table** based on the **numbers you have received**. **Note** that if the **first number is bigger than the** **second** one, you have to **print** the following message:

"**Try with other numbers.**"

**Otherwise**, you have to **print the multiplication table** in the following format:

**{num1} \* {num2} = {result}**

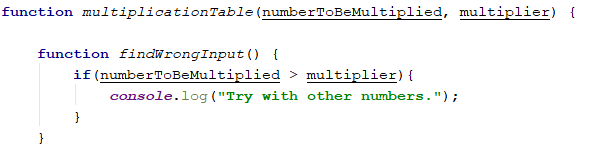
For more information, see the examples below:

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2, 9 | 2 \* 9 = 18  3 \* 9 = 27  4 \* 9 = 36  5 \* 9 = 45  6 \* 9 = 54  7 \* 9 = 63  8 \* 9 = 72  9 \* 9 = 81 |
| **Input** | **Output** |
| 8, 3 | Try with other numbers. |

### Hints

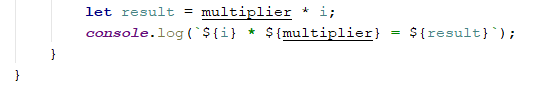
* The **first thing** we could do is **check** if **the first input number is bigger than the second one** by using a simple **if-statement** in a new **function**. For example, we could name it FindWrongInput().



* Next, we create another **function** where the table will be **printed**. In order to form a multiplication table, we need to **create a for-loop**. Its **start value** should be the **first input number** (**i = numberToBeMultiplied**) and it should be **executed until i <= multiplier**.

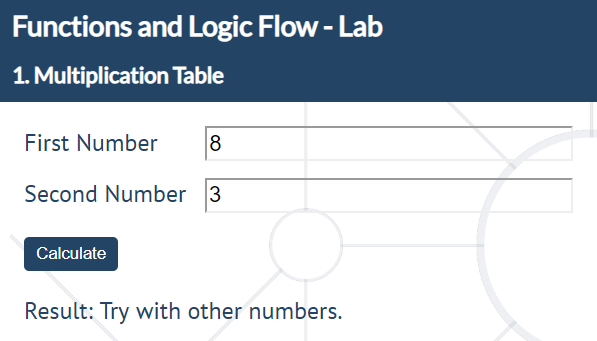


* **For each iteration**, we need to **create a variable** that stores the value of the **product** of the **multiplier by the current number**. Then we simply print the **current values of our variables**.



* However, we **must not forget to call the functions** that we just created in our **main function**, so that they can be **executed**.





## Temperature Converter

Write a JS function that receives **two arguments** as an input.

* **The** **first parameter** will be the **degrees** that need to be converted.
* **The second parameter** will be either **Fahrenheit** or **Celsius** (**case-insensitive**). Every other type of input is considered **invalid** and the following message should be printed: **"Error!"**

The **output** should consist of **one number** - the **converted** degrees, in case of a **valid** input. Note that you need to **round the degrees to the nearest integer**.

**Otherwise**, you should just print "**Error!**"

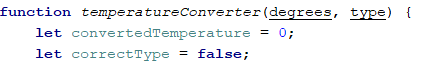
For more information, see the example below:

### Example

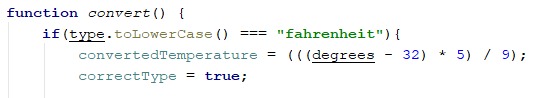
|  |  |
| --- | --- |
| **Input** | **Output** |
| 79, 'celsius' | 174 |
| **Input** | **Output** |
| 45, 'Fahrenheit' | 7 |
| **Input** | **Output** |
| 15, 'gosho' | Error! |

### Hints

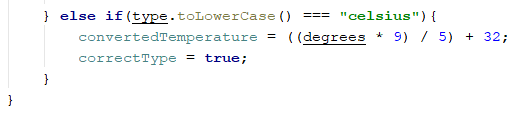
* **The first thing** we can do **is create a variable** that will **store** the **converted temperature**. Its initial value will be 0. We can also create a **bool** variable that will keep track if the given type of degrees is correct.



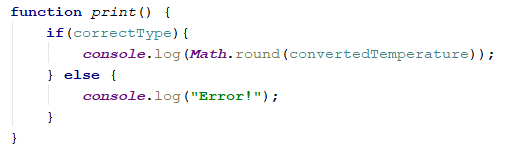
* **Next**, we create a **new function** where we check if the **type of temperature** we have received is a **correct** one. In order to do so, we can use an **if-else** conditional statement. **Note** that we have to consider that the input is **case-insensitive**. The method **toLowerCase()** is perfect for such conditions. We check if the given type is Fahrenheit and then we use the correct formula (google it, if you do not know it) for converting the degrees. We also set our **bool** variable to **true** because we will need it when printing the output.



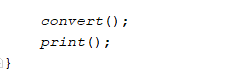
* Inside the same function, we write **exactly the same** method **as described above** in order to find if the **type of temperature** we have received is **Celsius**.

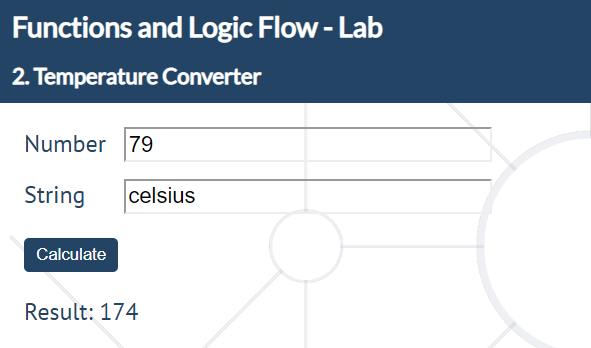


* After that, we can create **another function** where the output will be **printed**. Inside it, we check if the **type of temperature** is **correct** and **print** the **corresponding output**. Note that we use the method **Math.round()** to **round the degrees to the nearest integer**.



* Finally, we **call** our **nested function** inside the **main** one, so that they can be **executed**.





## Count Occurrences of a Given Character

Write a JS function that takes **two parameters** as an input.

* **The first parameter** will be a **string**.
* **The second parameter** will be a **character**.

Your task is to find **each occurrence** of the **character** in the string and **print** if the **total count** is **even** or **odd** in the following format:

**Count of ${char} is even/odd.**

There will **always** be **at least one** char **occurrence** in the string.

For more information, see the examples below:

### Example

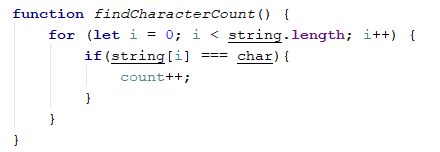
|  |  |
| --- | --- |
| **Input** | **Output** |
| 'HoHoHoHo Merry Crisis', 'H' | Count of H is even. |
| **Input** | **Output** |
| 'Is this real life?', 'f' | Count of f is odd. |

### Hints

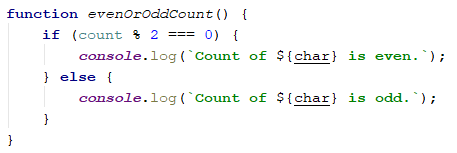
* We start by **declaring** a new **variable** that will keep track of the **count** of the **char occurrences**.



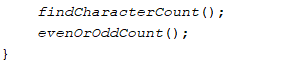
* Next, we create a **new function** where we have to **go through each** of the **characters** that the **given string** contains. We do that by using a **for-loop**. In the loop, we check if the **current character** is **equal** to the one we are **looking for**. If it is, we **increment count** by **one**.

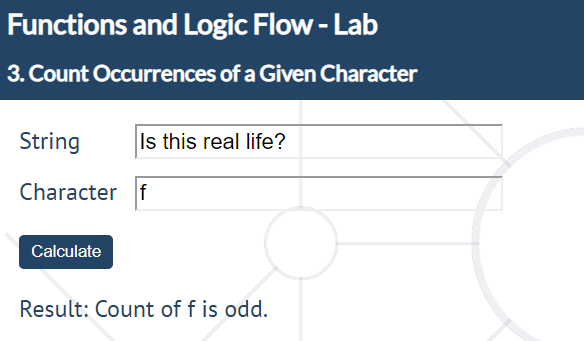


* After we have received the total count, the only thing we have to do is check if it is an **even** or an **odd** number. We do that by using the **modulus** operator. Then, we **print** the **corresponding** output.



* Finally, we **call** our **nested function** inside the **main** one, so that they can be **executed**.





## Unique Characters

Write a JS function that takes **one string parameter** as an input.

Your task is to **extract** only the **unique characters** from the string **except for whitespaces**.

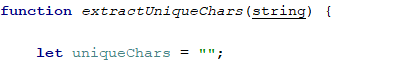
For more information, see the examples below:

### Example

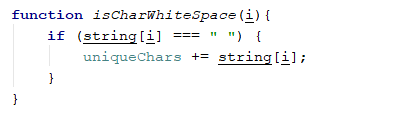
|  |  |
| --- | --- |
| **Input** | **Output** |
| 'Doggos are FunnNn' | Dogs are FunN |
| **Input** | **Output** |
| 'This is a random Sentence' | This a rndom Setc |

### Hints

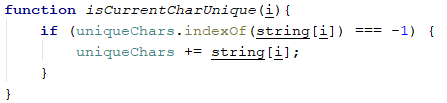
* We start off by creating a **new variable** that will be used to **store only the unique characters** from the string we have received.



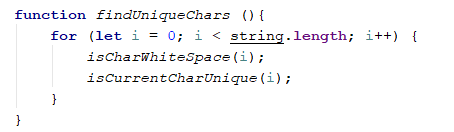
* Then, we start **looping** through **each** of the **characters** of our string with a **for-loop**. In the for-loop, we have to check if the **current character** is a **whitespace** and if it is, we have to **concatenate** it to the **variable** we just created. However, we can do that in a **new function** that will be created **out** of the loop but **executed** in it.



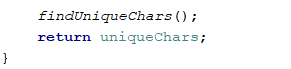
* After we have done that, in order to check if the **current character** is **unique or not**, we can use the method **indexOf()**. We have to check if **indexOf(string[i]) equals -1** because that means that the **current character is not present** in our string of special chars, therefore, **we have to append it**. Again, we can do that in a **new function** that will be created **out** of the loop but **executed** in it.

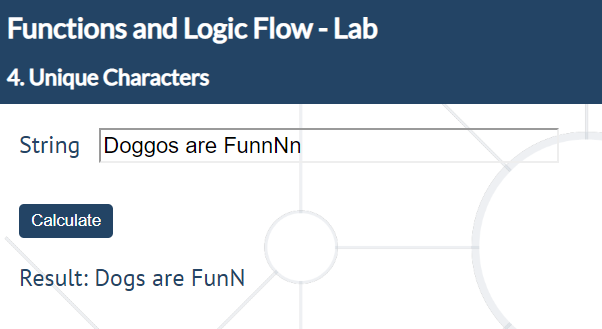


* The for-loop should look something like this:



* Finally, we call our function findUniqueChars() in the main one. By doing that, the two functions in the for-loop will be executed automatically. Then we return the result;





## Special Words

Write a JS function that receives **five parameters** as an input.

* The **first and the second parameter** will be **integers**.
* The **third, fourth and fifth** will be **strings**.

Your task is to **iterate from the first parameter to the second** one.

* For **numbers** which are **multiples** of **both three and five**, print **all three strings separated by a space**.
* For **multiples only of three**, print the **second string** (the **fourth** input **parameter**).
* For **multiples only of five**, print the **third string** (the **fifth** input **parameter**).

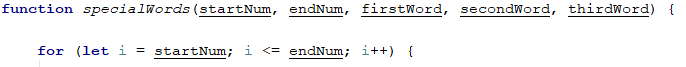
For more information, see the examples below:

### Example

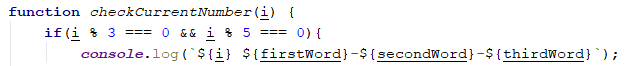
|  |  |
| --- | --- |
| **Input** | **Output** |
| 9, 15, "doggo", "pesho", "test" | 9 pesho  10 test  11  12 pesho  13  14  15 doggo-pesho-test |

### Hints

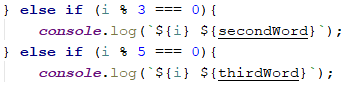
* The first thing we have to do is create a **for-loop** that **starts** from the **first input parameter** and is **executed** until the **second** one.



* We have to check if the **current number % 3** and the **current number % 5** equals 0. If it does, we print the **corresponding** output. We can separate our logic in a new function that will be created out of the loop but called in it.



* By using the **same** logic, we **separately** check for **multiples only of three** and **multiples only of five**.



* If **none** of the **conditions** are **true**, we print the **current number**.



* Finally, we call out function in the for-loop.



