Lab: Syntax, Functions and Statements

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/2749/Syntax-Functions-and-Statements-Lab

1. Echo Function

Write a JS function that takes **one string parameter** and **prints** on two lines the **length** of the parameter and then the **unchanged parameter** itself.

Examples

Input	Output
`'Hello, JavaScript!'`	`18` `Hello, JavaScript!`
`'strings are easy'`	`16` `strings are easy`

Hints

- Write a function that receives a single **parameter**.
- Use the console.log function to print text on the console. Each call prints a newline automatically.
- The string's **`length` property** is used to determine how many characters are in a given string

2. String Length

Write a JS function that takes **three string arguments** as an input. Calculate the **sum** of the **length** of the **strings** and the **average length** of the strings **rounded down** to the nearest integer.

The **input** comes as **three string arguments** passed to your function.

The **output** should be printed on the console in two lines.

Examples

Input	Output
`'chocolate', 'ice cream', 'cake'`	`22` `7`
`'pasta', '5', '22.3'`	`10` `3`

Hints

- Write a function that receives three string arguments.
- Declare two variables named **`sumLength`** and **`averageLength`** that will keep the mathematical results.
- Calculate the length of the strings using the **`length` property**.

```
function solve(arr1, arr2, arr3) {
    let sumLength;
    let averageLength;

    let firstArgumentLength = arr1.length;
    let secondArgumentLength = arr2.length;
    let thirdArgumentLength = arr3.length;
```

- Calculate the sum of the three lengths.

```
sumLength = firstArgumentLength + secondArgumentLength + thirdArgumentLength;
```

- Calculate the **average length** of the strings **rounded down** to the nearest integer. Use the **`Math.floor()`** function.

```
averageLength = Math.floor(sumLength / 3);
```

- Print the results on the console.

```
console.log(sumLength);
console.log(averageLength);
```

3. Largest Number

Write a function that takes **three number arguments** as input and finds the **largest** of them. Print the following text on the console: **`` `The largest number is {number}.` ``**.

The **input** comes as **three number arguments** passed to your function.

The **output** should be printed to the console.

Example

Input	Output
`5, -3, 16`	`The largest number is 16.`
`-3, -5, -22.5`	`The largest number is -3.`

Hints

- Write a function that receives three number arguments.
- Declare a variable named **`result`** that will keep the result.

```
function solve(num1, num2, num3) {
    let result;
}
```

- Make several checks to find out the largest of the three numbers. Start with num1.

```
if (num1 > num2 && num1 > num3) {
    result = num1;
}
```

- Do the same for the others.

```
else if (num2 > num1 && num2 > num3) {
    result = num2;
}
else if (num3 > num1 && num3 > num2) {
    result = num3;
}
```

- Print the result on the console.

```
console.log(`The largest number is ${result}.`)
```

4. Circle Area

Write a function that takes **a single argument** as an input. **Check the type** of input argument. If it is a **number**, assume it is the radius of a circle and **calculate the circle area**. Print the **area rounded** to **two decimal places**.

If the argument type is **NOT** a number, print the following text on the console:
`We can not calculate the circle area, because we receive a {type of argument}.`

The **input** comes as a **single argument** passed to your function.

The **output** should be printed on the console.

Example

Input	Output
5	78.54
'name'	We can not calculate the circle area, because we receive a string.

Hints

- Write a function that receives a single argument.
- Declare a variable named **result** that will keep your result.

```
function solve(input) {
    let result;
}
solve(5);
solve('name');
```

- Check the type of the input argument with the **typeof** operator.

```
let inputType = typeof(input);
```

- If the type is equal to **'number'**, calculate the circle area and print it on the console rounded to two decimal places. To do this, use the method **toFixed().**
  The **Math.pow()** function returns the base to the exponent power, that is, base exponent. You can find more information about the area here:

```
if (inputType === 'number') {
    result = Math.pow(input, 2) * Math.PI;
    console.log(result.toFixed(2));
}
```

- If the type is **NOT** a **'number'**, print the following text on the console:

```
else {
    console.log(`We can not calculate the circle area,
     because we receive a ${inputType}.`)
}
```

5. Math Operations

Write a JS function that takes **two numbers** and **a string** as an input.

The string may be one of the following: '**+**', '**-**', '*****', '**/**', '**%**', '******'.

Print on the console the result of the mathematical **operation** between **both numbers** and the **operator** you receive as a string.

The **input** comes as **two numbers** and **a string argument** passed to your function.

The **output** should be printed on the console.

Examples

Input	Output
5, 6, '+'	11
3, 5.5, '*'	16.5

Hints

- Write a function which receives **three** arguments:

```
function solve(num1, num2, operator) {


}
solve(5, 6, '+');
```

- Declare a variable named **result** that will keep your mathematical result.
- Write down the **switch** command that will take the string from your input and depending on it, perform the mathematical logic between the two numbers.

```
function solve(num1, num2, operator) {

    let result;
    switch (operator){
        case '+': result = num1+num2; break;
        case '-': result = num1-num2; break;
        case '/': result = num1/num2; break;
        case '*': result = num1*num2; break;
        case '%': result = num1%num2; break;
        case '**': result = num1**num2; break;
    }
    console.log(result);
}
```

- Print the result on the console.

```
console.log(result);
```

6. Sum of Numbers N…M

Write a JS function that takes two numbers **n** and **m** as an input and **prints the sum** of all numbers from **n** to **m**.

The **input** comes as **two string elements** that need to be **parsed** as numbers.

The **output** should **`return`** the **sum**.

Examples

Input	Output
`'1', '5'`	`15`
`'-8', '20'`	`174`

Hints

- Write a function that receives two string arguments and parse them as numbers. Use **`Number(string)`** function to parse the input.

```
function solve(n, m) {
    let num1 = Number(n);
    let num2 = Number(m);
}
```

- Declare a variable named **`result`** that will keep the mathematical results.
- Write a **`for`** loop from **`num1`** to **`num2`** and every turn of the cycle, until it's completed, add the current value.

```
for (let i = num1; i <= num2; i++) {
    result += i;
}
```

- Finally, return the result.

```
return result;
```

7. Day of Week

Write a function that prints a number between 1 and 7 when a **day of the week** is passed to it as a string and an **error message** if the string is **not recognized**.

The **input** comes as a single-string argument.

The **output** should be returned as a result.

Examples

Input	Output
`'Monday'`	`1`
`'Friday'`	`5`
`'Invalid'`	`error`

8. Days in a month

Write a JavaScript function to get the number of days in a month.

The input comes as two numeric parameters. The first element is the month, the second is the year.

The output must return the number of days in a month for a given year.

Examples

Input	Output
`1, 2012`	`31`
`2, 2021`	`28`

Hints

- Use **Date()**

9. Square of Stars

Write a function that **prints a rectangle** made of **stars** with variable **`size`**, depending on an input parameter. If there is **no parameter** specified, the rectangle should **always** be of **size 5**. Look at the examples to get an idea.

The **input** comes as a single **number** argument.

The **output** is a series of lines printed on the console, forming a rectangle of variable size.

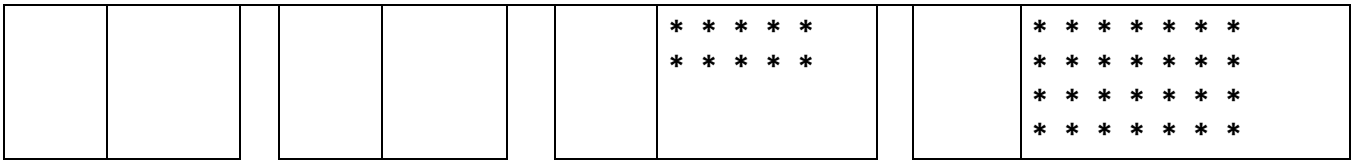
Examples

Input	Output
1	*

Input	Output
2	* * * *

Input	Output
5	* * * * * * * * * * * * * * *

Input	Output
7	* *



10. Aggregate Elements

Write a program that performs different operations on an array of elements. Implement the following operations:

- **Sum(a_i)** - calculates the sum of all elements from the input array
- **Sum($1/a_i$)** - calculates the sum of the inverse values ($1/a_i$) of all elements from the array
- **Concat(a_i)** - concatenates the string representations of all elements from the array

The **input** comes as an array of number elements.

The **output** should be printed on the console on a new line for each of the operations.

Examples

Input	Output

`[1, 2, 3]`	`6`
	`1.8333333333333333`
	`123`

Input	Output

`[2, 4, 8, 16]`	`30`
	`0.9375`
	`24816`