JavaScript

JavaScript add behaviour to webpage. It improves the user experience and make webpage more interactive.

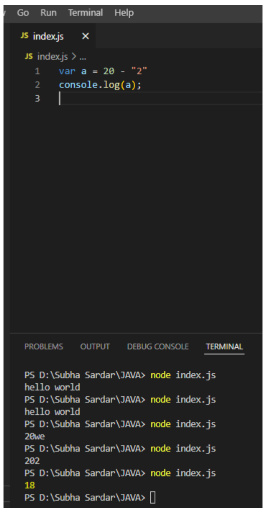
- **Value & Variable:**

var name = “subha sardar”;

variable(key) | variable name | value

- **DATA TYPES IN JAVASCRIPT**

Si Data Types that are primitives:
undefined: typeof instance === "undefined"
Boolean:typeof instance === "boolean"
Number: typeof instance === "number"
String: typeof instance === "string"
Bigint: typeof instance === "bigint"
Symbol: typeof instance === "symbol"

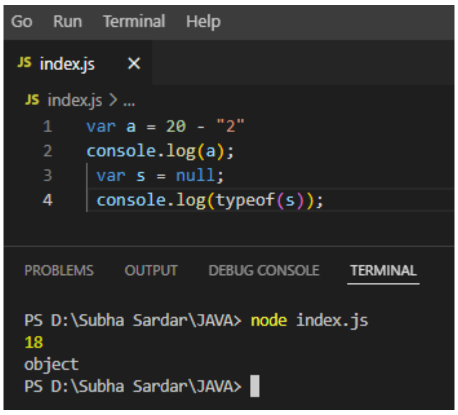


```
var a = 20 - "2"
console.log(a);
```

```
PS D:\Subha Sardar\JAVA> node index.js
hello world
PS D:\Subha Sardar\JAVA> node index.js
hello world
PS D:\Subha Sardar\JAVA> node index.js
20we
PS D:\Subha Sardar\JAVA> node index.js
202
PS D:\Subha Sardar\JAVA> node index.js
18
PS D:\Subha Sardar\JAVA> 
```

Here is one BUG in JavaScript on ‘-’ or subtract function.

Another bug in JavaScript on null value:



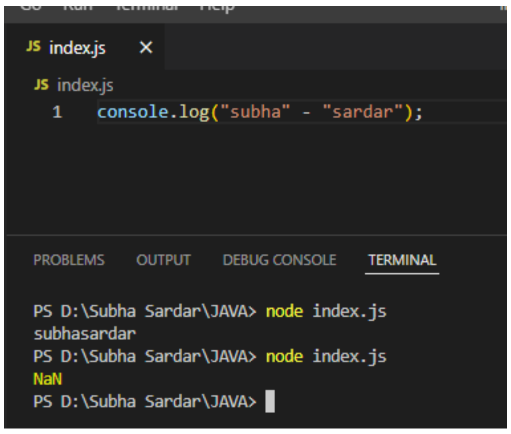
```javascript
var a = 20 - "2"
console.log(a);
var s = null;
console.log(typeof(s));
```

```
PS D:\Subha Sardar\JAVA> node index.js
18
object
PS D:\Subha Sardar\JAVA>
```

What is NaN?

NaN is a property of the global object.

Not a Number



```javascript
console.log("subha" - "sardar");
```

```
PS D:\Subha Sardar\JAVA> node index.js
subhasardar
PS D:\Subha Sardar\JAVA> node index.js
NaN
PS D:\Subha Sardar\JAVA>
```

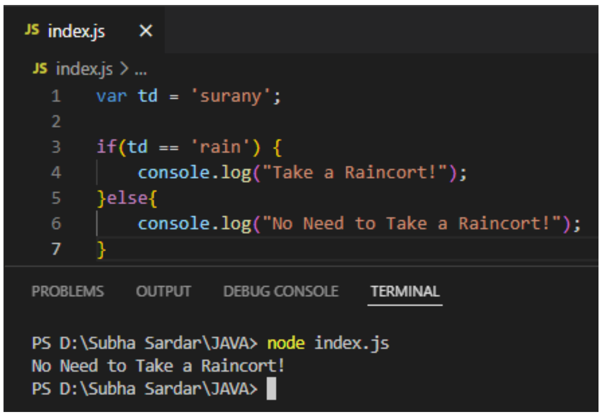
➢ **EXPRESSIONS AND OPERATORS**

- Assignment operators (=)
- Arithmetic operators (+, -, /, *, %)
- Comparison operators (==, !=, <, >, >=, <=)
- Logical operators (&&, ||, !)
- String operators (Concatenation `+`)
- Conditional (ternary) operator

Note: == check only value and === check value & data type

➢ **CONTROL STATEMENT & LOOPS**

- If..Else



```javascript
var td = 'surany';

if(td == 'rain') {
    console.log("Take a Raincort!");
}else{
    console.log("No Need to Take a Raincort!");
}
```

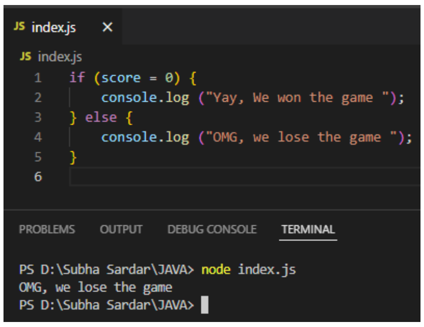
```
PS D:\Subha Sardar\JAVA> node index.js
No Need to Take a Raincort!
PS D:\Subha Sardar\JAVA>
```

- Switch Statement
- While Loop
- Do-While Loop
- For Loop
- For in Loop
- For of Loop
- Conditional (ternary) operator

What are truthy and falsy values in JavaScript?

Total 5 falsy values in JavaScript
0,"", undefined, null, NaN, false** is false anyway

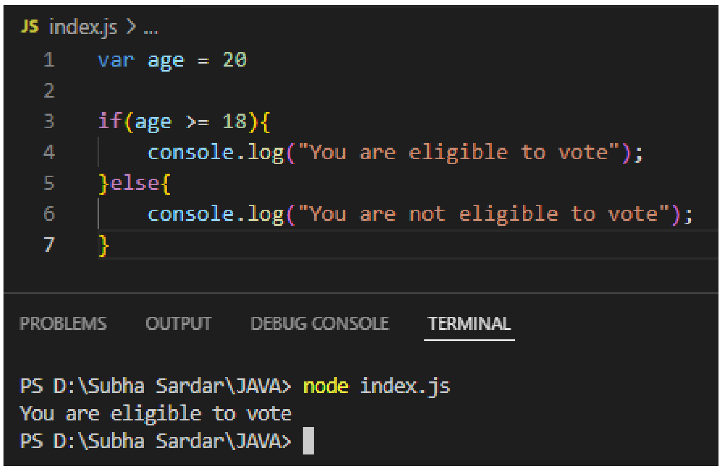
```javascript
if (score = 0) {
   console.log ("Yay, We won the game ");
} else {
   console.log ("OMG, we lose the game ");
}
```



```javascript
if (score = 0) {
    console.log ("Yay, We won the game ");
} else {
    console.log ("OMG, we lose the game ");
}
```

```
PS D:\Subha Sardar\JAVA> node index.js
OMG, we lose the game
PS D:\Subha Sardar\JAVA>
```

If..else condition:

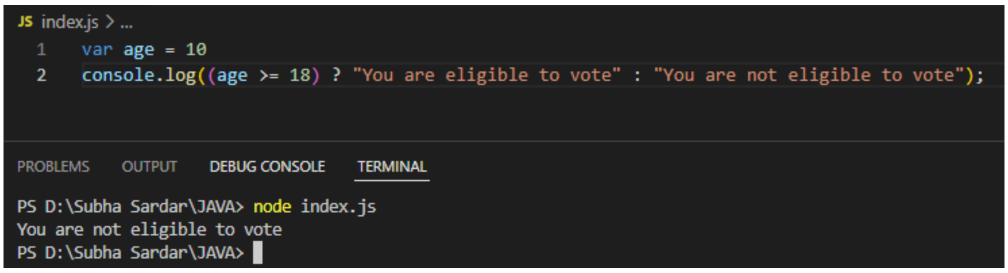


index.js > ...

```
var age = 20

if(age >= 18){
    console.log("You are eligible to vote");
}else{
    console.log("You are not eligible to vote");
}
```

```
PS D:\Subha Sardar\JAVA> node index.js
You are eligible to vote
PS D:\Subha Sardar\JAVA> 
```

Conditional (ternary) operator:

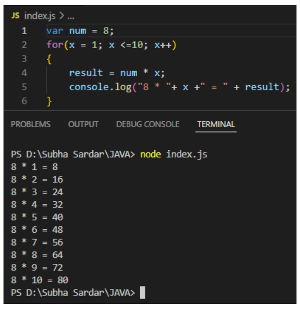


index.js > ...

```
var age = 10
console.log((age >= 18) ? "You are eligible to vote" : "You are not eligible to vote");
```

```
PS D:\Subha Sardar\JAVA> node index.js
You are not eligible to vote
PS D:\Subha Sardar\JAVA> 
```

➢ **Loop: (While, do-while & for)**



index.js > ...

```
var num = 8;
for(x = 1; x <=10; x++)
{
    result = num * x;
    console.log("8 * "+ x +" = " + result);
}
```

```
PS D:\Subha Sardar\JAVA> node index.js
8 * 1 = 8
8 * 2 = 16
8 * 3 = 24
8 * 4 = 32
8 * 5 = 40
8 * 6 = 48
8 * 7 = 56
8 * 8 = 64
8 * 9 = 72
8 * 10 = 80
PS D:\Subha Sardar\JAVA> 
```

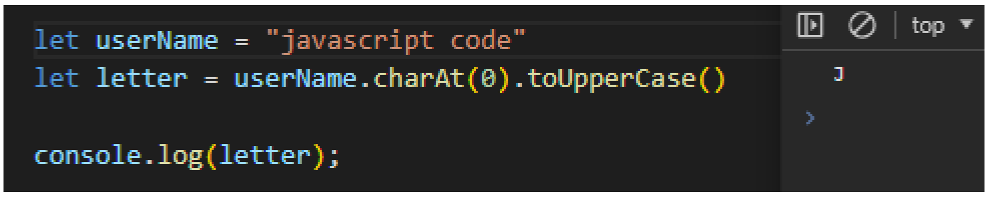
➢ **Function** is reusable and use many times with new arguments.

What is Function Parameter and Function Argument?

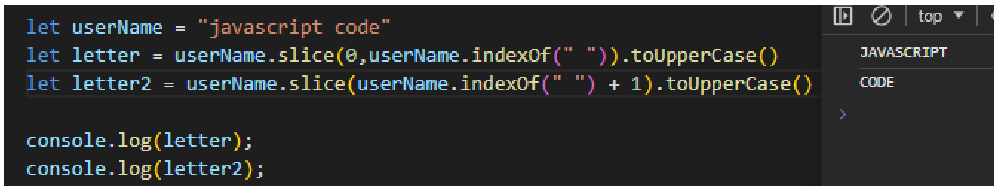
```
Function sum (a, b)    // a and be are parameter
{
  Var total = a+b;
  Console.log(total);
}
Sum()
Sum (10,20)   // 10 and 20 are argument
```

Method chaining:

Calling one method after another in one continuous line of code.


```
let userName = "javascript code"
let letter = userName.charAt(0).toUpperCase()

console.log(letter);
```

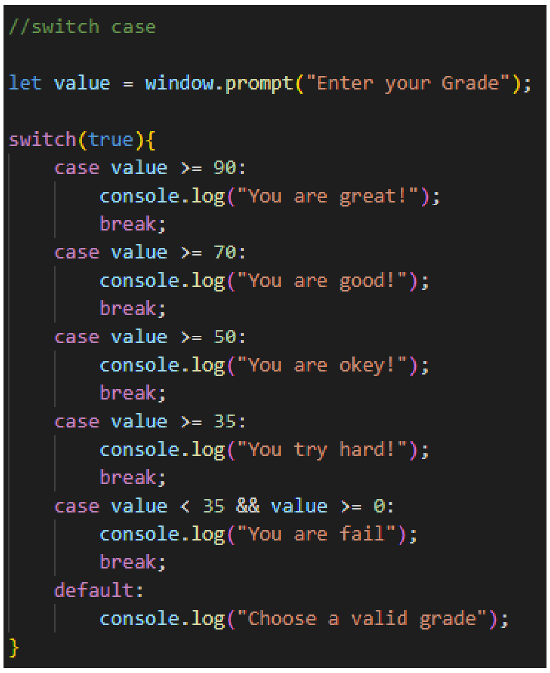
```
J
```


```
let userName = "javascript code"
let letter = userName.slice(0,userName.indexOf(" ")).toUpperCase()
let letter2 = userName.slice(userName.indexOf(" ") + 1).toUpperCase()

console.log(letter);
console.log(letter2);
```

```
JAVASCRIPT
CODE
```

Switch Case:

Chaining of multiple condition & Operator user --


```
//switch case

let value = window.prompt("Enter your Grade");

switch(true){
    case value >= 90:
        console.log("You are great!");
        break;
    case value >= 70:
        console.log("You are good!");
        break;
    case value >= 50:
        console.log("You are okey!");
        break;
    case value >= 35:
        console.log("You try hard!");
        break;
    case value < 35 && value >= 0:
        console.log("You are fail");
        break;
    default:
        console.log("Choose a valid grade");
}
```

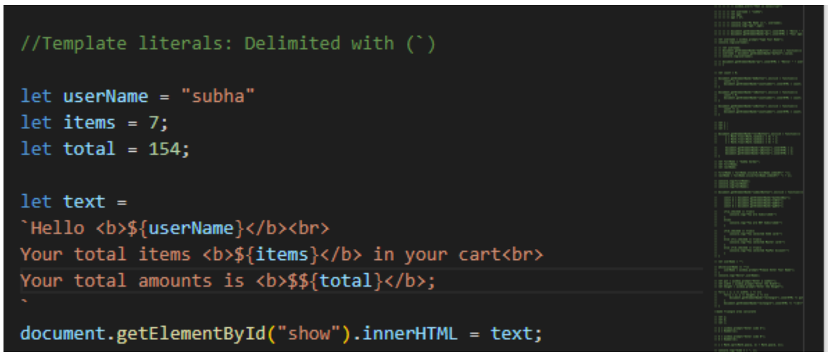
Variable scope:

Where a variable is accessible.

Let = variables are limited to block scope {}
Var = variables are limited to a function () {}
Global variable = is declared outside of any function.

Template literals:

Delimited with (`)

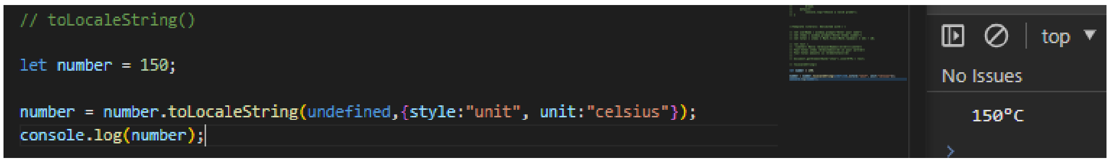


```
//Template literals: Delimited with (`)

let userName = "subha"
let items = 7;
let total = 154;

let text =
`Hello <b>${userName}</b><br>
Your total items <b>${items}</b> in your cart<br>
Your total amounts is <b>$${total}</b>;
`
document.getElementById("show").innerHTML = text;
```

Hello **subha**
Your total items **7** in your cart
Your total amounts is **$154**;

toLocaleString()

Return a string with a language sensitive representation of number.
Ex: toLocaleString(locale, {options});

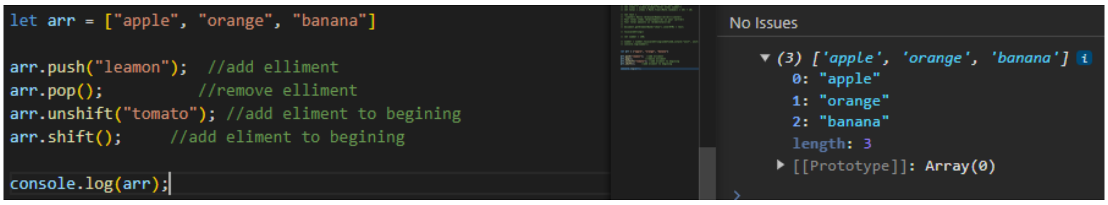


```
// toLocaleString()

let number = 150;

number = number.toLocaleString(undefined,{style:"unit", unit:"celsius"});
console.log(number);
```

```
150°C
```

➢ **Array**

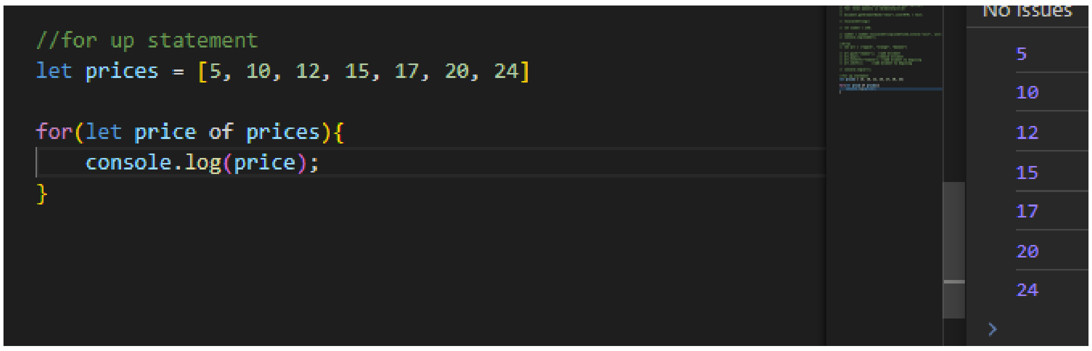
Like this a variable that can store multiple values.



```
let arr = ["apple", "orange", "banana"]

arr.push("leamon");  //add elliment
arr.pop();           //remove elliment
arr.unshift("tomato"); //add eliment to begining
arr.shift();      //add eliment to begining

console.log(arr);
```

```
▼ (3) ['apple', 'orange', 'banana']
    0: "apple"
    1: "orange"
    2: "banana"
    length: 3
  ▶ [[Prototype]]: Array(0)
```

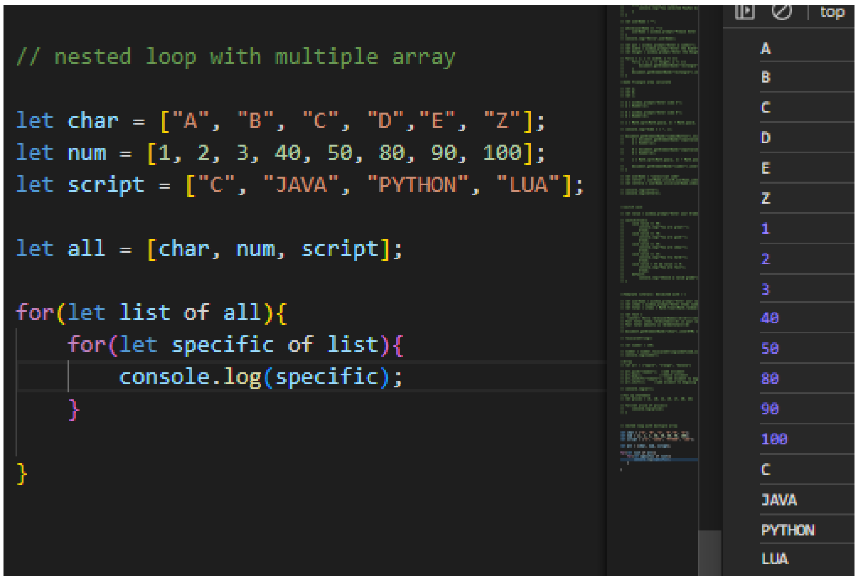
for-up-statement (another type of for loop uses)



```
//for up statement
let prices = [5, 10, 12, 15, 17, 20, 24]

for(let price of prices){
    console.log(price);
}
```

```
5
10
12
15
17
20
24
```

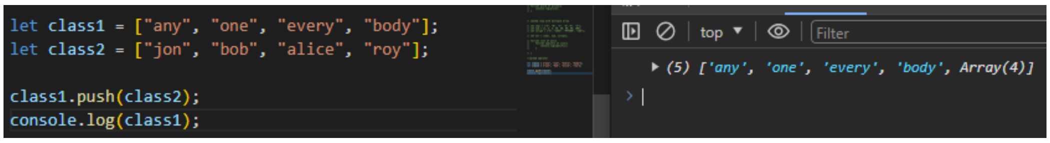
Nested loop with multiple array:



```
// nested loop with multiple array

let char = ["A", "B", "C", "D","E", "Z"];
let num = [1, 2, 3, 40, 50, 80, 90, 100];
let script = ["C", "JAVA", "PYTHON", "LUA"];

let all = [char, num, script];

for(let list of all){
    for(let specific of list){
        console.log(specific);
    }

}
```

```
A
B
C
D
E
Z
1
2
3
40
50
80
90
100
C
JAVA
PYTHON
LUA
```

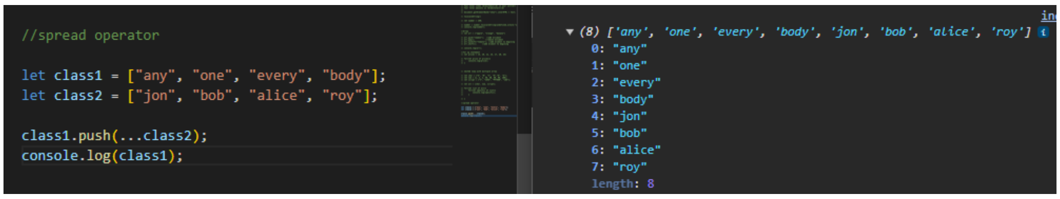
Spread operator:
Allow an inerrable such as an array or string to be expended in place where Zero or more arguments are expected (unpack the elements).



```
let class1 = ["any", "one", "every", "body"];
let class2 = ["jon", "bob", "alice", "roy"];

class1.push(class2);
console.log(class1);
```

```
▶ (5) ['any', 'one', 'every', 'body', Array(4)]
```

Add spread operator (...)



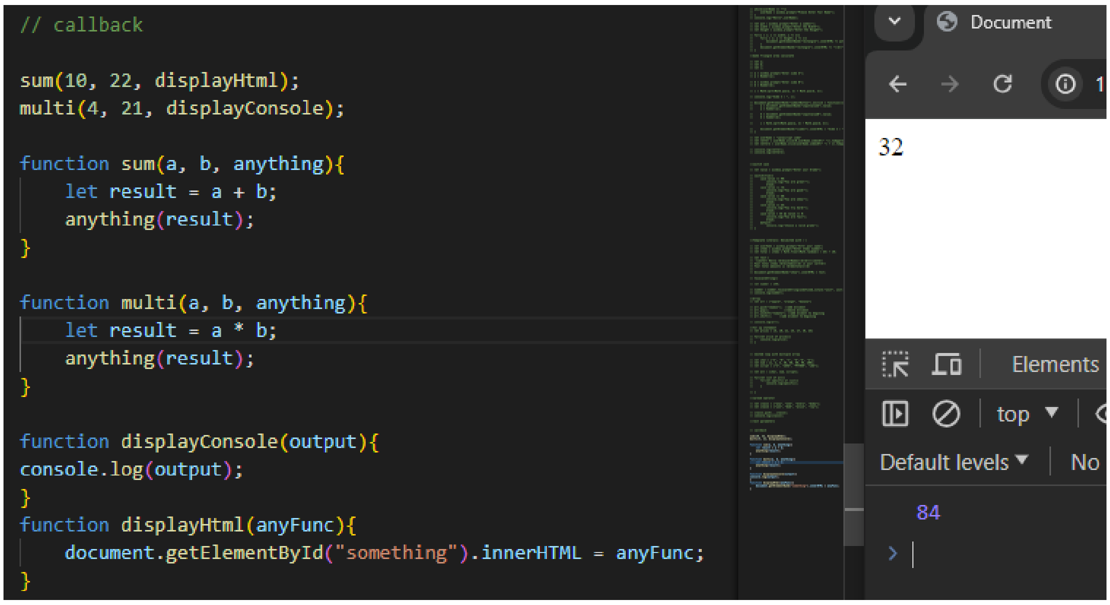
```
//spread operator

let class1 = ["any", "one", "every", "body"];
let class2 = ["jon", "bob", "alice", "roy"];

class1.push(...class2);
console.log(class1);
```

```
▼ (8) ['any', 'one', 'every', 'body', 'jon', 'bob', 'alice', 'roy']
    0: "any"
    1: "one"
    2: "every"
    3: "body"
    4: "jon"
    5: "bob"
    6: "alice"
    7: "roy"
    length: 8
```

Rest parameters:
Represents an indefinite number of parameters.
(Packs arguments into an array)

Callback:

A function passed as an argument to another function. Ensure that a function is not going to run before a task is completed. Help us develop asynchronous code. When one function has to wait for another function. That helps us to avoid errors and protentional problems.
Ex: wait for a file to load.



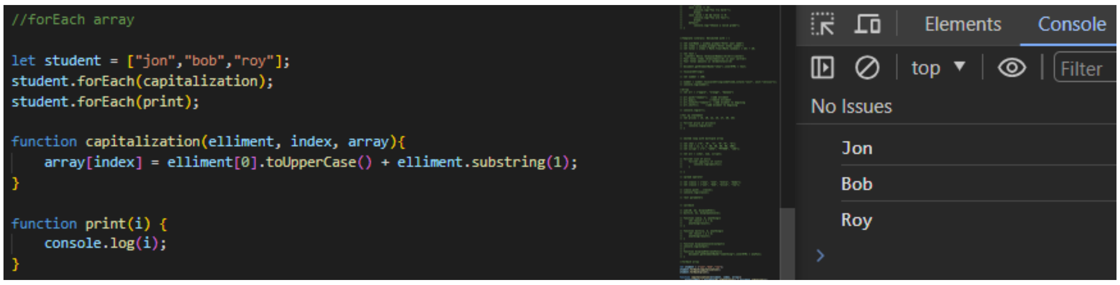
```
// callback

sum(10, 22, displayHtml);
multi(4, 21, displayConsole);

function sum(a, b, anything){
    let result = a + b;
    anything(result);
}

function multi(a, b, anything){
    let result = a * b;
    anything(result);
}

function displayConsole(output){
console.log(output);
}
function displayHtml(anyFunc){
    document.getElementById("something").innerHTML = anyFunc;
}
```

```
32
```

```
84
```

forEach:

Execute a provided callback function once for each array element.

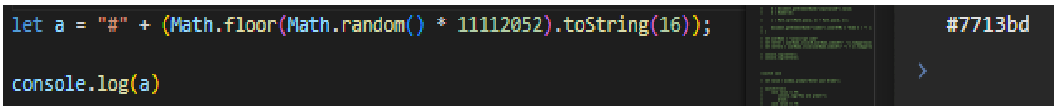


```
//forEach array

let student = ["jon","bob","roy"];
student.forEach(capitalization);
student.forEach(print);

function capitalization(elliment, index, array){
    array[index] = elliment[0].toUpperCase() + elliment.substring(1);
}

function print(i) {
    console.log(i);
}
```

```
Jon
Bob
Roy
```

For random color || make a number to string || Hex string

```
let a = "#" + (Math.floor(Math.random() * 11112052).toString(16));
console.log(a)
```

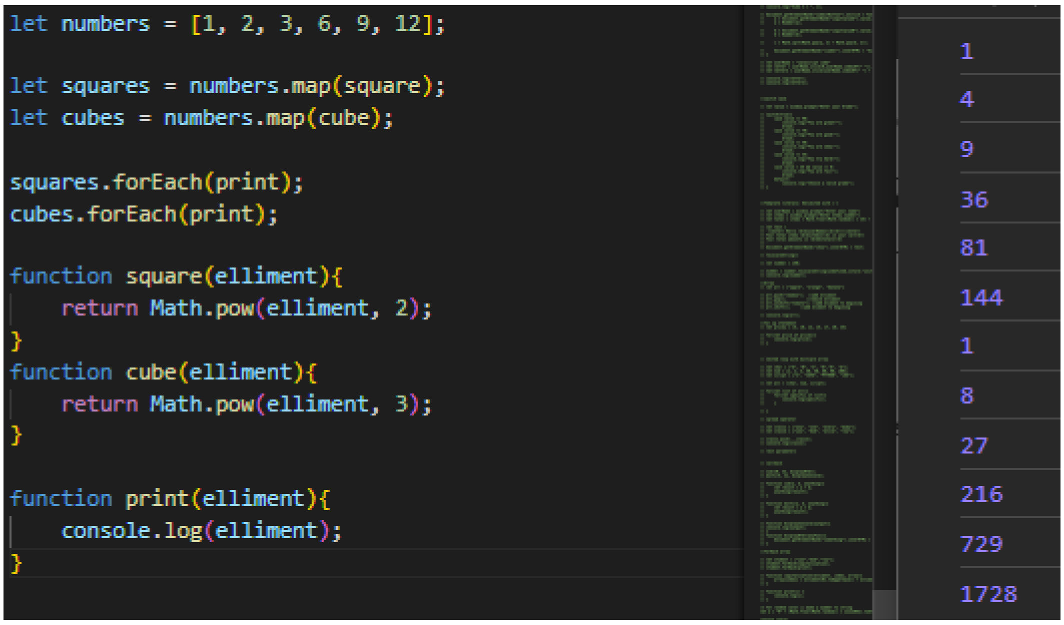


```
let a = "#" + (Math.floor(Math.random() * 11112052).toString(16));

console.log(a)
```

```
#7713bd
```

array.map():

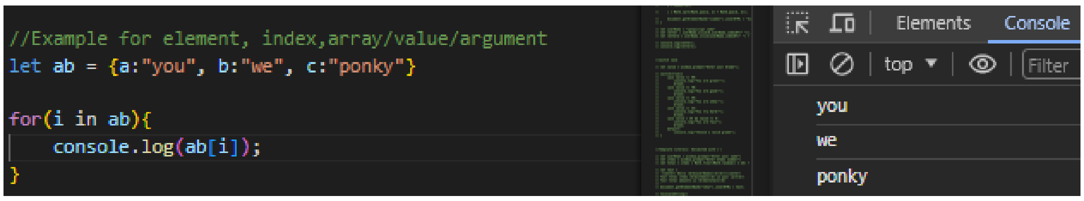
Execute a provided callback function once for each array element and create a new array.



```
let numbers = [1, 2, 3, 6, 9, 12];

let squares = numbers.map(square);
let cubes = numbers.map(cube);

squares.forEach(print);
cubes.forEach(print);

function square(elliment){
    return Math.pow(elliment, 2);
}
function cube(elliment){
    return Math.pow(elliment, 3);
}

function print(elliment){
    console.log(elliment);
}
```

```
1
4
9
36
81
144
1
8
27
216
729
1728
```

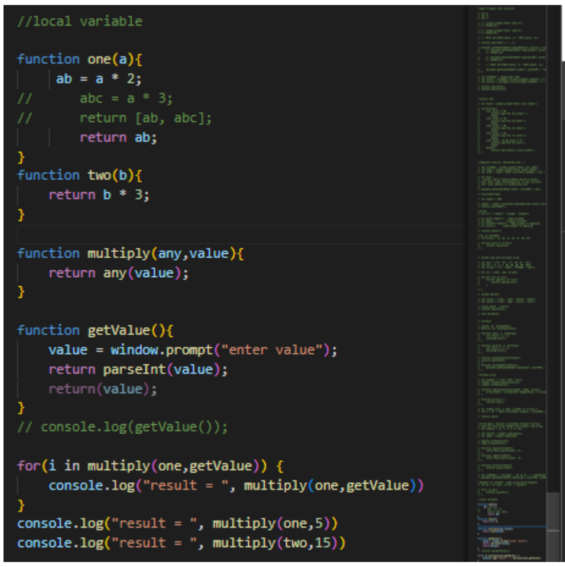
For loop with list, index and array:



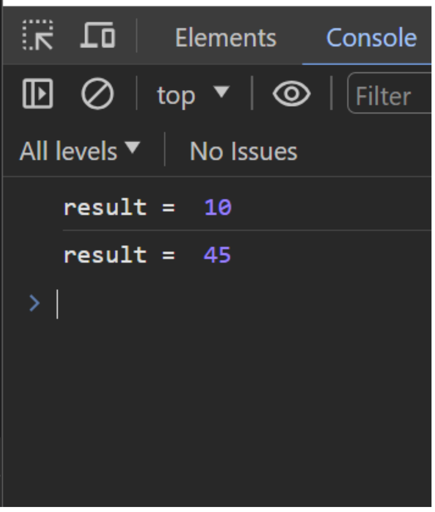
```
//Example for element, index,array/value/argument
let ab = {a:"you", b:"we", c:"ponky"}

for(i in ab){
    console.log(ab[i]);
}
```

```
you
we
ponky
```

local variable:



```
//local variable

function one(a){
    ab = a * 2;
//      abc = a * 3;
//      return [ab, abc];
        return ab;
}
function two(b){
    return b * 3;
}

function multiply(any,value){
    return any(value);
}

function getValue(){
    value = window.prompt("enter value");
    return parseInt(value);
    return(value);
}
// console.log(getValue());

for(i in multiply(one,getValue)) {
    console.log("result = ", multiply(one,getValue))
}
console.log("result = ", multiply(one,5))
console.log("result = ", multiply(two,15))
```



```
result =  10
result =  45
>
```