

# Array Methods:

## Arrays:-

Array is data structure used to store multiple values of any data type sequentially.

## Features of ~~an~~ array:-

1. Order collection
2. Homogeneous or heterogeneous (mixed - datatypes)
3. Mutable
4. Dynamic size
5. Multi dimensional Array.

## Creating Arrays:-

We can create an array using square brackets [] and separating the elements with commas.

Ex:- `let arr = [1, 2, 'html', 'css']`

## Accessing Elements:-

You can access elements of an array using square brackets and the index of the element. Remember that array indexes start at 0.

Ex:-  
`console.log(arr[0]); // 1`  
`console.log(arr[3]); // css`

## Modifying Elements:-

You can modify elements in an array by assigning a new value to a specific index.

Ex:-  
`arr[1] = 'js'`  
`console.log(arr); // [1, 'js', 'html', 'css']`

dynamic size:-

we can increase array size dynamically

Ex:-  
`var arr = ["hello", "world", 2];`  
`arr[3] = 3;`  
`console.log(arr);` // ["hello", "world", 2, 3]

Multidimensional Array:-

Ex:-  
`var arr = [1, 2, 3, [4, 5, [6, 7, 8]]];`  
`console.log(arr[3][2][2]);`

Array Methods:-

JavaScript provides many built-in methods to work with arrays such as push, pop, shift, unshift, slice, splice, concat, indexOf, includes, and many more.

1. Array length:-

length property returns the number of elements in an Array

Ex:-  
`let arr = [1, 2, "html", "css"]`  
`console.log(arr.length);` // 4

2. Array at():-

The `at()` method of Array instance takes an integer value and returns the item at the index, allowing for positive and negative integers. Negative integers count back from the last item of an array.

\* Point last index value of array (length-1)

Print: `console.log(arr[arr.length-1]);`

Example:-

`let arr = [1, 2, 3, 4];`  
`console.log(arr.at(0));` // 1  
`console.log(arr.at(-1));` // 4

3. Concat():- method is used to merge two ~~arrays~~ (or more arrays). It does not modify the existing arrays but instead returns a new array containing the elements of original arrays concatenated together.

Ex:-

```
var arr1 = [1, 2, 3]
```

```
var arr2 = ['a', 'b', 'c']
```

```
console.log(arr1.concat(arr2)); // [1, 2, 3, 'a', 'b', 'c']
```

• Logps in arrays:-

```
var a = [1, 2, 3, 4, 5, 6, 7, 8]
```

```
for (i=0; i<a.length; i++){  
    console.log(a[i]);  
}
```

```
for (i in a) {  
    console.log(i);  
}
```

4. Array Splice():-

splice() method is used to change the contents of an array by removing or replacing existing elements and/or adding new elements in place.

• It modifies the original array and return an array containing the removed elements.

Syntax:-

```
array.splice(startIndex, endIndex, add item 1, add item 2);
```

Example:- // adding at specific index.

```
var arr = ["html", "css", "js", "react"]
```

```
arr.splice(1, 0, "bootstrap");
```

```
console.log(arr); // [html, bootstrap, css, js, react]
```



// replacing at a specific index

```
arr.splice(1, 1, "bootstrap");
```

```
console.log(arr); // [html, bootstrap, js, react]
```

// replacing all values at a time

```
arr.splice(0, 4, "java", "python", "nodejs", "mysql");
```

```
console.log(arr); // [java, python, nodejs, mysql]
```

### 5. Array Slice:-

slice() method is used to extract section of an array and returns a new array containing the extracted elements.

Syntax:- `array.slice(startIndex, endIndex);`

Example:- `var arr = ["html", "css", "js", "react"]`

```
var arr2 = arr.slice(0, 2);
```

```
console.log(arr2); // [html, css]
```

```
var arr2 = arr.slice(-1);
```

```
console.log(arr2); // [react]
```

### 6. Array Pop:-

Pop method removes the last element from an array and returns the element.

### 7. Array Push:-

Push method adds one or more elements to the end of an array and returns new length of array.

Example:-

```
var arr = ['html', 'css', 'js', 'react'];
```

```
arr.pop(); // ['html', 'css', 'js']
```

```
arr.push("git");
```

```
console.log(arr); // [html, css, js, git]
```

### 8. Array Shift();

shift method() is used to remove the first element of an array and returns that element.

### 9. Array unshift();

unshift method adds one or more elements to the beginning of an array and return new length of an array.

Example:-

```
var arr = ["html", "css", "js", "React"];  
arr.shift(); // ["css", "js", "React"]  
arr.unshift("git"); // ["git", "html", "css", "js", "React"]
```

### 10. Array Sort and Reverse Methods:-

```
arr.sort();
```

```
arr.reverse();
```

sort  
Examples:-

```
var arr = [9, 3, 4, 67];
```

```
console.log(arr.sort());
```

Output:- 3, 4, 67, 9 [Lexographical order it takes only first letter of an element]

Reverse Example:-

```
var arr = ["html", "css", "js"];
```

```
console.log(arr.reverse()); // ["js", "css", "html"]
```

Problem:- reverse an array and push into the new array

```
var a = ["html", "css", "js", "React"]  
var new = []
```

```
for (i = a.length - 1; i >= 0; i--) {
```

```
  console.log(a[i]);
```

```
  new.push(a[i]);
```

```
}
```

```
console.log(new); // ["React", "js", "css", "html"]
```

## TASKS:

1) Add elements to an array and iterate using for loop and for in loop and for of loop  
Create an array of your favorite movies and iterate an array to the console.

The screenshot shows a VS Code editor with a file named `iterating_array.html`. The code defines an array `arr` with values `[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]` and iterates over it using three different loops: a standard `for` loop, a `for...in` loop, and a `for...of` loop. The browser console on the right shows the output of these iterations, displaying the array and its elements.

```
2 <html lang="en">
8 <body>
9 <!--
10 1) Add elements to an array and iterate using for loop
11 Create an array of your favorite movies and iterate
12 -->
13
14 <script>
15   var arr=[1,2,3,4,5,6,7,8,9,10]
16   for(i=0;i<arr.length; i++){
17   }
18   console.log(arr);
19
20   for(i in arr){
21     // console.log(i);
22     console.log(arr[i]);
23   }
24
25   for(i of arr){
26     console.log(i);
27   }
28 </script>
29
```

2) Remove elements from an array  
Remove the first and last elements from the array.

The screenshot shows a VS Code editor with a file named `removing_elements_from_array.html`. The code defines an array `arr` with values `[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`, logs it, and then removes the first element using `arr.shift()` and the last element using `arr.pop()`. The browser console on the right shows the output, displaying the array before and after each removal operation.

```
12 <!--
13
14
15 <script>
16   var arr=[1,2,3,4,5,6,7,8,9,10]
17   console.log("given array "+arr);
18
19   arr.shift();
20   console.log("removed first element "+arr);
21   arr.pop();
22   console.log("removed last element "+arr);
23
24
```

3) Reverse an array using for loop  
Hints: use push method

The screenshot shows a VS Code editor with a file named `reverse_array.html`. The code defines an array `arr` with values `['html', 'css', 'js', 'react', 'git']` and iterates over it in reverse order using a `for` loop, pushing each element into a new array `rev`. The browser console on the right shows the output, displaying the reversed array.

```
2 <html lang="en">
3 <head>
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <title>Document</title>
6 </head>
7 <body>
8 <!--
9 3) Reverse an array using for loop
10 hints
11 use push method
12 -->
13
14 <script>
15   var arr=['html','css','js','react','git']
16   var rev=[]
17   for(i=arr.length-1;i>=0;i--){
18     rev.push(arr[i]);
19   }
20   console.log(rev);
21
22
```

4) find the even and odd numbers in an array [12,3,5,6,22,56,29]  
and print the even numbers array and the sum of add and odd numbers array and the sum of even

The screenshot shows a web browser with a JavaScript program running. The program iterates through an array [12, 3, 5, 6, 22, 56, 29] and identifies even and odd numbers. The console output is as follows:

```

12 is even number
6 is even number
22 is even number
56 is even number
96 is the total count of even numbers
3 is odd number
5 is odd number
29 is odd number
37 is the total count of odd numbers

```

5) Take a heterogeneous array and separate each data type into new array  
hints : use loop, typeof and push method  
inp: let arr = ["apple", "banana", "mango", "banana", 3, 4, 5, 6, true, {name: "object"}];  
out :  
num=[3,4,5,6]  
str=["apple","banana","mango","banana"]  
bool=[true]  
obj=[{name: "object"}]

The screenshot shows a web browser with a JavaScript program running. The program takes a heterogeneous array and separates its elements into four new arrays based on their data type. The console output is as follows:

```

[3, 4, 5, 6]
["apple", "banana", "mango", "banana"]
[true]
[{name: "object"}]

```