

# JavaScript **Array** Methods

push()

sort()

indexOf()

pop()

includes()

lastIndexOf()

shift()

slice()

reverse()

unshift()

map()

concat()

find()

filter()

join()

some()

reduce()

toString()

every()

forEach()

# push()

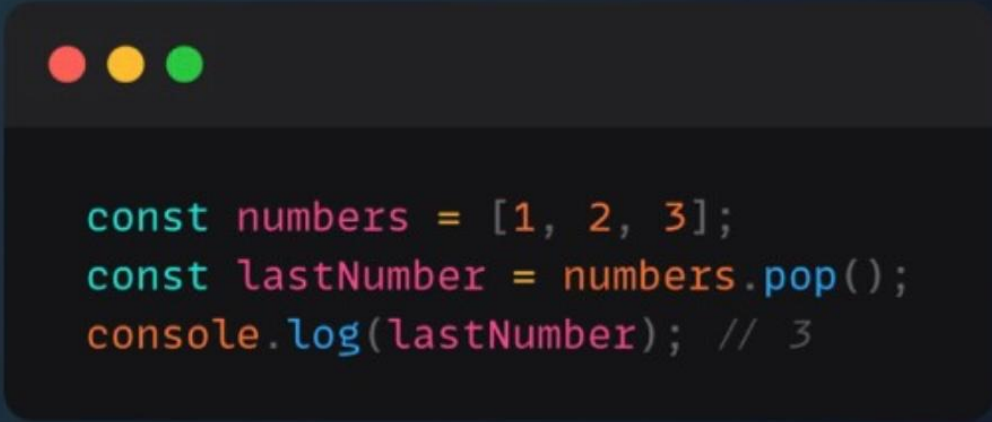
Adds one or more elements to the **end** of an array and returns the new length of the array..



```
const numbers = [1, 2, 3];  
numbers.push(4, 5);  
console.log(numbers);  
// [1, 2, 3, 4, 5]
```

# pop()

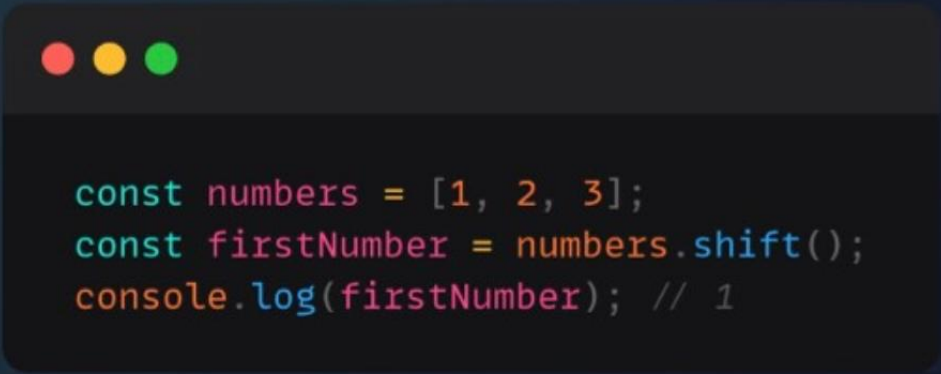
Removes the last element from an array and returns that element.



```
const numbers = [1, 2, 3];  
const lastNumber = numbers.pop();  
console.log(lastNumber); // 3
```

# shift()

Removes the **first** element from an array and returns that element.



```
const numbers = [1, 2, 3];  
const firstNumber = numbers.shift();  
console.log(firstNumber); // 1
```

# unshift()


**Adds** one or more elements to the **beginning** of an array and returns the new length of the array.



```
const numbers = [1, 2, 3];  
numbers.unshift(0, -1);  
console.log(numbers); // [0, -1, 1, 2, 3]
```

# find()

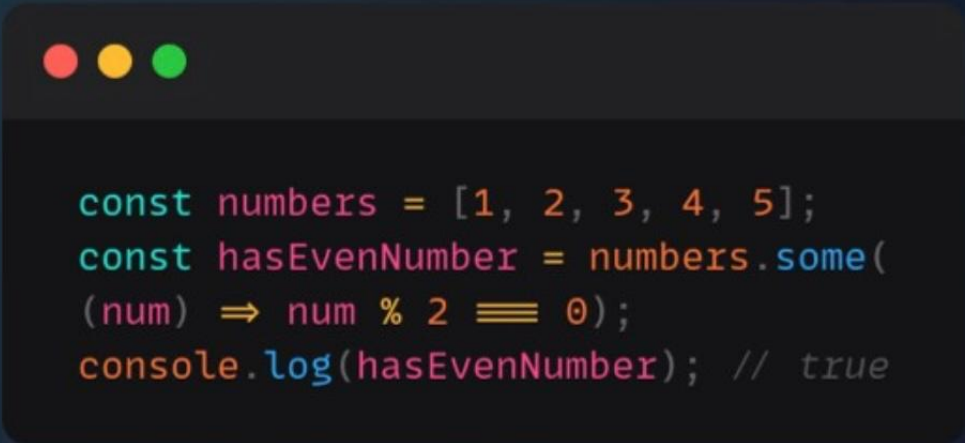
Returns the value of the **first** element in the array that **satisfies** the provided testing function. Otherwise, undefined is returned.



```
const numbers = [1, 2, 3, 4, 5];  
const foundNumber = numbers.find((num) => num > 3);  
console.log(foundNumber); // 4
```

# some()

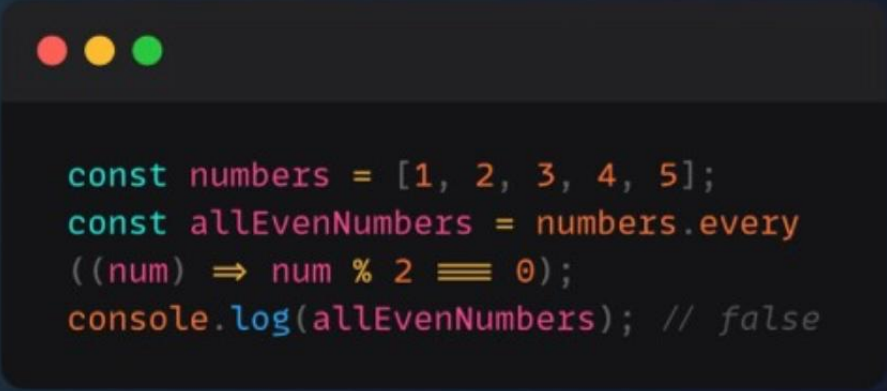
Tests whether **at least one** element in the array passes the **test** implemented by the provided function. It returns true if any element passes the test, otherwise it returns false.



```
const numbers = [1, 2, 3, 4, 5];  
const hasEvenNumber = numbers.some(  
  (num) => num % 2 === 0);  
console.log(hasEvenNumber); // true
```

# every()

Tests whether **all** elements in the array pass the **test** implemented by the provided function. It returns true if all elements pass the test, otherwise it returns false.



```
const numbers = [1, 2, 3, 4, 5];  
const allEvenNumbers = numbers.every  
((num) => num % 2 === 0);  
console.log(allEvenNumbers); // false
```



# sort()

Sorts the elements of an array in place and returns the **sorted** array. The default sort order is built upon converting the elements into strings, then comparing their sequences of UTF-16 code units values.

```
const fruits = ['banana', 'apple', 'orange', 'grape'];  
fruits.sort();  
console.log(fruits);  
// ['apple', 'banana', 'grape', 'orange']
```

```
const numbers = [100, 20, 200, 30];  
numbers.sort((a, b) => a - b);  
console.log(numbers); // [20, 30, 100, 200]
```

# includes()

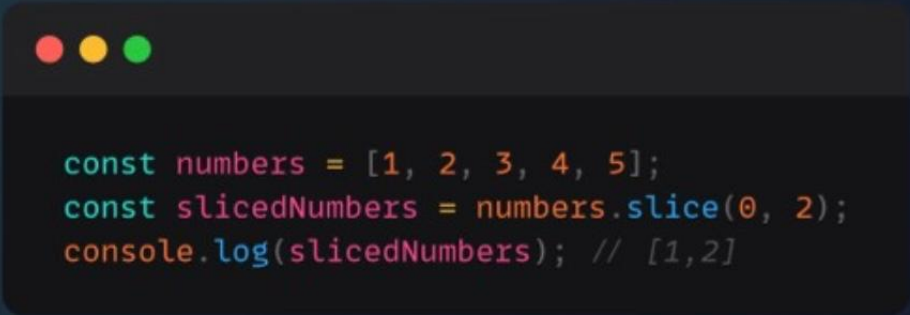
Determines **whether** an array **includes** a certain element, returning true or false as appropriate.



```
const numbers = [1, 2, 3, 4, 5];  
const includesThree = numbers.includes(3);  
console.log(includesThree); // true
```

# slice()

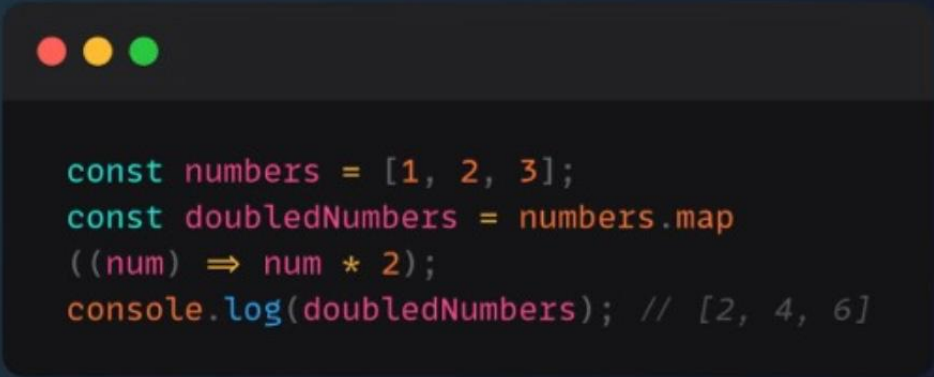
Returns a shallow copy of a **portion** of an array into a **new array** object selected from start to end (**end not included**). The original array will not be modified.



```
const numbers = [1, 2, 3, 4, 5];  
const slicedNumbers = numbers.slice(0, 2);  
console.log(slicedNumbers); // [1,2]
```

# map()

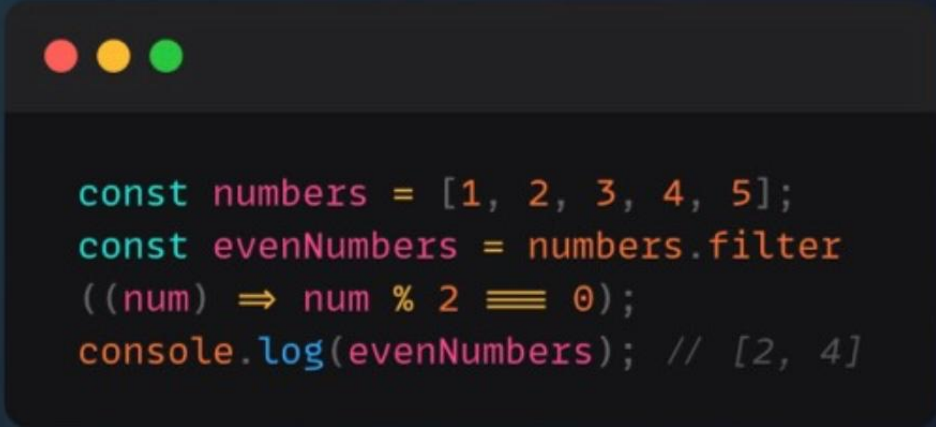
Creates a **new** array with the results of calling a provided function on **every** element in the calling array.



```
const numbers = [1, 2, 3];  
const doubledNumbers = numbers.map  
((num) => num * 2);  
console.log(doubledNumbers); // [2, 4, 6]
```

# filter()

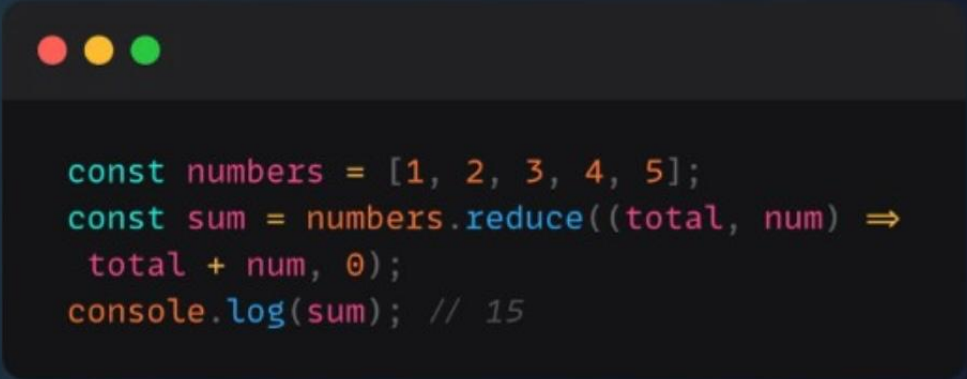
Creates a **new** array with all elements that **pass** the test implemented by the provided function.



```
const numbers = [1, 2, 3, 4, 5];  
const evenNumbers = numbers.filter  
((num) => num % 2 === 0);  
console.log(evenNumbers); // [2, 4]
```

# reduce()

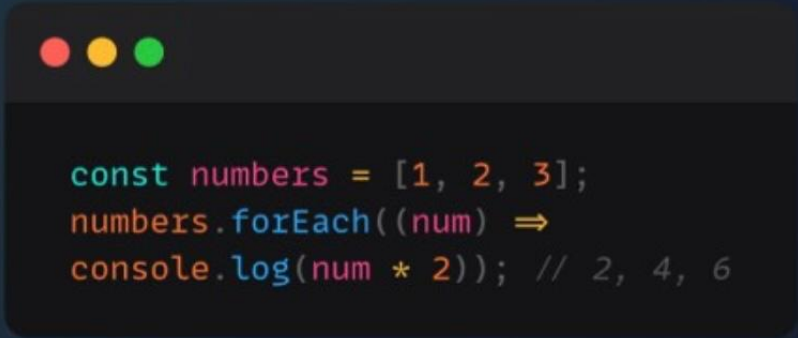
Executes a reducer function on each element of the array, resulting in a **single output value**.



```
const numbers = [1, 2, 3, 4, 5];  
const sum = numbers.reduce((total, num) =>  
  total + num, 0);  
console.log(sum); // 15
```

# forEach()


Executes a provided function once for each array element.



```
const numbers = [1, 2, 3];  
numbers.forEach((num) =>  
  console.log(num * 2)); // 2, 4, 6
```

# indexOf()

Returns the **first index** at which a given element can be found in the **array**, or -1 if it is not present.



```
const fruits =  
  ['banana', 'apple', 'orange', 'grape'];  
const appleIndex = fruits.indexOf('apple');  
console.log(appleIndex); // 1
```



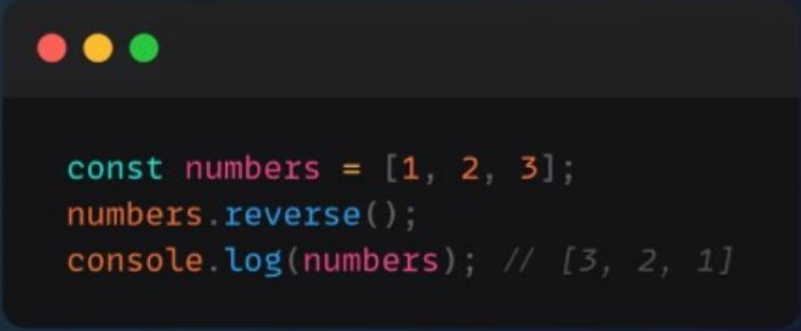
# lastIndexOf()

Returns the **last index** at which a given element can be found in the **array**, or -1 if it is not present.

```
const fruits =  
  ['banana', 'apple', 'orange', 'grape', 'apple'];  
const lastAppleIndex = fruits.lastIndexOf('apple');  
console.log(lastAppleIndex); // 4
```

# reverse()

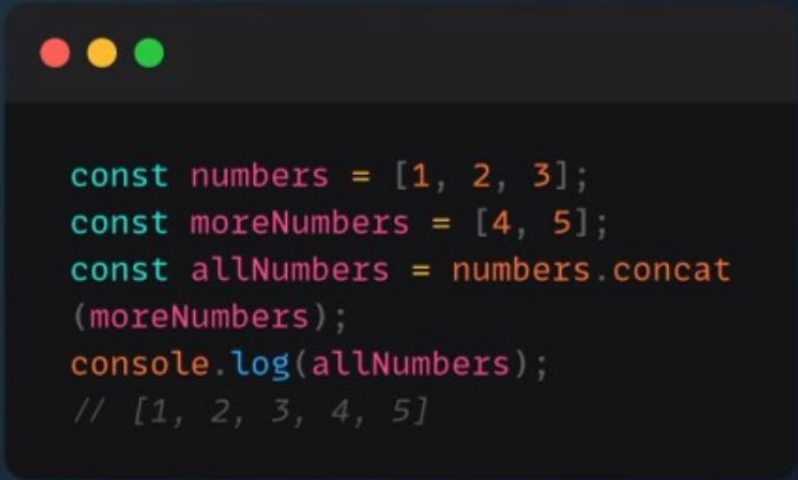
Reverses the order of the elements of an array in place. The first element becomes the last, and the last element becomes the first.



```
const numbers = [1, 2, 3];  
numbers.reverse();  
console.log(numbers); // [3, 2, 1]
```

# concat()

Returns a **new array** that includes elements from the original array and additional elements.



```
const numbers = [1, 2, 3];  
const moreNumbers = [4, 5];  
const allNumbers = numbers.concat  
(moreNumbers);  
console.log(allNumbers);  
// [1, 2, 3, 4, 5]
```

# join()

**Joins** all elements of an array into a **string**.  
The elements are separated by a specified **separator** string.

```
const fruits =  
  ['banana', 'apple', 'orange', 'grape'];  
const joinedFruits = fruits.join(', ');  
console.log(joinedFruits);  
// 'banana, apple, orange, grape'
```

# toString()

Returns a **string** representing the specified number or array and its elements.



```
const numbers = [1, 2, 3];  
const numbersString = numbers.toString();  
console.log(numbersString); // '1, 2, 3'
```