JAVASCRIPT

ARRAY METHODS

20 JavaScript
Array Methods
You need to know

Sonu Madheshiya

1. map( ) : This method creates a new array with the results of calling a provided function on every element in this array.

```
const numbers = [1, 2, 3, 4];
const doubled = numbers.map(number => number * 2);
console.log(doubled); // [2, 4, 6, 8]
```

Sonu Madheshiya

2. filter( ): This method creates a new array with only elements that pass the condition inside the provided function.

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

Sonu Madheshiya

3. sort( ): This method is used to arrange/sort array's elements either in ascending or descending order.

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

Sonu Madheshiya

4. forEach( ): This method helps to loop over array by executing a provided callback function for each element in an array.

```
const numbers = [1, 2, 3, 4];
numbers.forEach(number => console.log(number));
// Output:
// 1
// 2
// 3
// 4
```

Sonu Madheshiya

5. concat( ): This method is used to merge two or more arrays and returns a new array, without changing the existing arrays.

```
const array1 = [1, 2, 3];
const array2 = [4, 5, 6];
const combined = array1.concat(array2);
console.log(combined); // [1, 2, 3, 4, 5, 6]
```

Sonu Madheshiya

6. every( ): This method checks every element in the array that passes the condition, returning true or false as appropriate.

```
const numbers = [1, 2, 3, 4];
const allPositive = numbers.every(number => number > 0);
console.log(allPositive); // true
```

Sonu Madheshiya

7. some( ): This method checks if at least one element in the array that passes the condition, returning true or false as appropriate.

```
const numbers = [1, 2, 3, 4];
const hasNegative = numbers.some(number => number < 0);
console.log(hasNegative); // false
```

Sonu Madheshiya

8. includes( ): This method checks if an array includes the specified element, returning true or false as appropriate.

```
const fruits = ['apple', 'banana', 'cherry'];
const hasApple = fruits.includes('apple');
console.log(hasApple); // true
```

Sonu Madheshiya

9. join( ): This method returns a new string by concatenating all of the array's elements separated by the specified separator.

```
const fruits = ['apple', 'banana', 'cherry'];
const fruitString = fruits.join(', ');
console.log(fruitString); // 'apple, banana, cherry'
```

Sonu Madheshiya

10. reduce( ): This method applies a function against an accumulator and each element in the array to reduce it to a single value.

```
const numbers = [1, 2, 3, 4];
const sum = numbers.reduce((acc, number) => acc + number, 0);
console.log(sum); // 10
```

Sonu Madheshiya

11. find( ): This method returns the value of the first element in an array that passes the test in a testing function.

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

Sonu Madheshiya

12. findIndex( ): This method returns the index of the first element in an array that passes the test in a testing function.

```
const numbers = [1, 2, 3, 4];
const index = numbers.findIndex(number => number > 2);
console.log(index); // 2
```

Sonu Madheshiya

13. indexOf( ): This method returns the index of the first occurrence of the specified element in the array, or -1 if it is not found.

```
const fruits = ['apple', 'banana', 'cherry'];
const index = fruits.indexOf('banana');
console.log(index); // 1
```

Sonu Madheshiya

14. fill( ): This method fills the elements in an array with a static value and returns the modified array.

```
const array = [1, 2, 3, 4];
array.fill(0);
console.log(array); // [0, 0, 0, 0]
```

Sonu Madheshiya

15. slice( ): This method returns a new array with specified start to end elements.

```
const fruits = ['apple', 'banana', 'cherry', 'date'];
const sliced = fruits.slice(1, 3);
console.log(sliced); // ['banana', 'cherry']
```

Sonu Madheshiya

16. reverse( ): This method reverses an array in place. Element at last index will be first and element at 0 index will be last.

```
const array = [1, 2, 3, 4];
array.reverse();
console.log(array); // [4, 3, 2, 1]
```

Sonu Madheshiya

17. push( ): This method adds one or more elements to the end of array and returns the new length of the array.

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

Sonu Madheshiya

18. pop( ): This method removes the last element from the end of array and returns that element.

```
const array = [1, 2, 3, 4];

const lastElement = array.pop();

console.log(array); // [1, 2, 3]

console.log(lastElement); // 4
```

Sonu Madheshiya

19. shift( ): This method removes the first element from an array and returns that element.

```
const array = [1, 2, 3, 4];
const firstElement = array.shift();
console.log(array); // [2, 3, 4]
console.log(firstElement); // 1
```

Sonu Madheshiya

20. unshift( ): This method adds one or more elements to the beginning of an array and returns the new length of the array.

```
const array = [1, 2, 3];
const newLength = array.unshift(0);
console.log(array); // [0, 1, 2, 3]
console.log(newLength); // 4
```

Sonu Madheshiya

Don't Forget to

Follow us!

Sonu Madheshiya

Sonu Madheshiya