TypeScript Array Methods

1. push()

Description: Adds one or more elements to the **end** of an array.
Syntax: array.push(element1, element2, ...)
Example:

```
let numbers = [1, 2, 3];

numbers.push(4, 5);

console.log(numbers); // Output: [1, 2, 3, 4, 5]
```

2. pop()

Description: Removes the **last** element from an array and returns it.
Syntax: array.pop()
Example:

```
let fruits = ['apple', 'banana', 'mango'];

let lastFruit = fruits.pop();

console.log(fruits); // Output: ['apple', 'banana']

console.log(lastFruit); // Output: 'mango'
```

3. shift()

Description: Removes the **first** element from an array and returns it.
Syntax: array.shift()
Example:

```
let numbers = [1, 2, 3];

let first = numbers.shift();

console.log(numbers); // Output: [2, 3]

console.log(first);   // Output: 1
```

4. unshift()

Description: Adds one or more elements to the **beginning** of an array.
Syntax: array.unshift(element1, element2, ...)
Example:

```
let fruits = ['banana', 'orange'];

fruits.unshift('apple');

console.log(fruits); // Output: ['apple', 'banana', 'orange']
```

5. concat()

Description: Combines two or more arrays and returns a **new** array.
Syntax: array.concat(value1, value2, ...)
Example:

```
let a = [1, 2];

let b = [3, 4];

let result = a.concat(b);

console.log(result); // Output: [1, 2, 3, 4]
```

6. slice()

Description: Extracts a **section** of an array without modifying the original array.
Syntax: array.slice(startIndex, endIndex)

Note: endIndex is **exclusive**.
Example:

```
let fruits = ['kiwi', 'apple', 'banana', 'mango'];

let selected = fruits.slice(1, 3);

console.log(selected); // Output: ['apple', 'banana']
```

7. splice()

Description: Adds or removes elements at any position in the array.
Syntax: array.splice(start, deleteCount, item1, item2, ...)
Example 1 – Remove:

```
let fruits = ['apple', 'banana', 'cherry'];

fruits.splice(1, 1);

console.log(fruits); // Output: ['apple', 'cherry']
```

Example 2 – Add:

```
fruits.splice(1, 0, 'orange');

console.log(fruits); // Output: ['apple', 'orange', 'cherry']
```

8. indexOf()

Description: Returns the **index** of the first occurrence of a value, or -1 if not found.
Syntax: array.indexOf(value, fromIndex?)
Example:

```
let fruits = ['apple', 'banana', 'cherry'];

console.log(fruits.indexOf('banana')); // Output: 1

console.log(fruits.indexOf('grape'));  // Output: -1
```

9. includes()

Description: Checks if an array **contains** a specific value. Returns true or false.
Syntax: array.includes(value, fromIndex?)
Example:

```
let fruits = ['apple', 'banana'];

console.log(fruits.includes('banana')); // Output: true

console.log(fruits.includes('grape'));  // Output: false
```

10. toString()

Description: Converts the array to a **comma-separated string**.
Syntax: array.toString()
Example:

```
let numbers = [1, 2, 3];

console.log(numbers.toString()); // Output: '1,2,3'
```

11. forEach()

Description: Executes a function **once** for each array element.
Syntax:

```
array.forEach(function(element, index, array) {...})
```

Example:

```
let fruits = ['apple', 'banana'];

fruits.forEach((fruit, i) => {

  console.log(`${i + 1}. ${fruit}`);

});

// Output:

// 1. apple

// 2. banana
```

12. map()

Description: Creates a **new array** with the result of a function applied to every element.
Syntax:

```
array.map(function(element, index, array) {...})
```

Example:

```
let nums = [1, 2, 3];

let squares = nums.map(n => n * n);

console.log(squares); // Output: [1, 4, 9]
```

13. filter()

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

```
array.filter(function(element, index, array) {...})
```

Example:

```
let nums = [1, 2, 3, 4];

let evens = nums.filter(n => n % 2 === 0);

console.log(evens); // Output: [2, 4]
```

14. reduce()

Description: Applies a function to reduce the array to a **single value** (e.g., sum).
Syntax:

```
array.reduce(function(accumulator, element, index, array), initialValue)
```

Example:

```
let nums = [1, 2, 3];

let total = nums.reduce((sum, n) => sum + n, 0);

console.log(total); // Output: 6
```

15. some()

Description: Returns true if **at least one** element passes the test.
Syntax:

```
array.some(function(element, index, array) {...})
```

Example:

```
let nums = [1, 2, 3];

let hasNegative = nums.some(n => n < 0);

console.log(hasNegative); // Output: false
```

16. every()

Description: Returns true if **all elements** pass the test.
Syntax:

```
array.every(function(element, index, array) {...})
```

Example:

```
let nums = [2, 4, 6];

let allEven = nums.every(n => n % 2 === 0);

console.log(allEven); // Output: true
```