Arrays

1. Array Creation and Accessing Elements Array Creation Arrays in JavaScript are used to store multiple values in a single variable. You can create arrays in several ways: • Using Array Literals:

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

• Using the Array Constructor:

```
let numbers = new Array(10); // Creates an array with 10 empty slots let colors = new
Array('red', 'green', 'blue'); // Creates an array with specified elements
```

Accessing Elements • Access array elements using their index (starting from `0`):

```
console.log(fruits[0]); // Output: apple console.log(fruits[2]); // Output: cherry
```

• Modify an element by assigning a new value:

```
fruits[1] = 'mango'; // Changes banana to mango
```

2. Common Array Methods push() • Adds one or more elements to the end of the array. • Returns the new length of the array.

```
let numbers = [1, 2, 3]; numbers.push(4); // [1, 2, 3, 4]
```

pop() • Removes the last element of the array. • Returns the removed element.

```
let numbers = [1, 2, 3, 4]; let last = numbers.pop(); // last = 4, numbers = [1, 2, 3]
```

shift() • Removes the first element of the array. • Returns the removed element.

```
let numbers = [1, 2, 3]; let first = numbers.shift(); // first = 1, numbers = [2, 3]
```

unshift() • Adds one or more elements to the beginning of the array. • Returns the new length of the array.

```
let numbers = [2, 3]; numbers.unshift(1); // [1, 2, 3]
```

3. Iterating Over Arrays forEach() • Executes a provided function once for each array element. • Does not return a new array.

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

map() • Creates a new array by applying a function to each element. • Does not modify the original array.

```
let numbers = [1, 2, 3]; let squares = numbers.map(num => num * num); // [1, 4, 9]
```

filter() • Creates a new array with elements that pass the provided condition (test). • Does not modify the original array.

```
let numbers = [1, 2, 3, 4]; let evens = numbers.filter(num => num % 2 === 0); // [2, 4]
```

Summary TableMethodPurposeModifies Original Array?Returns?**push()Add elements to the endYesNew array lengthpop()Remove the last elementYesRemoved elementshift()Remove the first elementYesRemoved elementunshift()Add elements to the beginningYesNew array lengthforEach()Iterate over elementsNoUndefinedmap()Create a new array with modified valuesNoNew arrayfilter()**Create a new array based on a conditionNoNew array

- Using Array Literals:

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

- Using the Array Constructor:

```
let numbers = new Array(10); // Creates an array with 10 empty slots
let colors = new Array('red', 'green', 'blue'); // Creates an array with specified elements
```

- Access array elements using their index (starting from `0`):

```
console.log(fruits[0]); // Output: apple
console.log(fruits[2]); // Output: cherry
```

- Modify an element by assigning a new value:

```
fruits[1] = 'mango'; // Changes banana to mango
```

- Adds one or more elements to the end of the array.

- Returns the new length of the array.

```
let numbers = [1, 2, 3];
numbers.push(4); // [1, 2, 3, 4]
```

- Removes the last element of the array.
- Returns the removed element.

```
let numbers = [1, 2, 3, 4];
let last = numbers.pop(); // last = 4, numbers = [1, 2, 3]
```

- Removes the first element of the array.
- Returns the removed element.

```
let numbers = [1, 2, 3];
let first = numbers.shift(); // first = 1, numbers = [2, 3]
```

- Adds one or more elements to the beginning of the array.
- Returns the new length of the array.

```
let numbers = [2, 3];
numbers.unshift(1); // [1, 2, 3]
```

- Executes a provided function once for each array element.
- Does not return a new array.

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

- Creates a new array by applying a function to each element.
- Does not modify the original array.

```
let numbers = [1, 2, 3];
let squares = numbers.map(num => num * num); // [1, 4, 9]
```

- Creates a new array with elements that pass the provided condition (test).
- Does not modify the original array.

```
let numbers = [1, 2, 3, 4];
let evens = numbers.filter(num => num % 2 === 0); // [2, 4]
```