Array Methods

Array.fill()

The fill() method changes all elements in an array to a static value, from a start index (default 0) to an end index (default array.length). It returns the modified array.

JavaScript Demo: Array.fill()

```
const array1 = [1, 2, 3, 4];

// fill with 0 from position 2 until position 4
console.log(array1.fill("$", 2, 4));
// expected output: [1, 2, "$", "$"]

// fill with 5 from position 1
console.log(array1.fill(5, 1));
// expected output: [1, 5, 5, 5]

console.log(array1.fill(6));
// expected output: [6, 6, 6, 6]

```

Array.find()

The find() method returns the first element in the provided array that satisfies the provided testing function. If no values satisfy the testing function, undefined is returned.

JavaScript Demo: Array.find()

```
const array1 = [5, 12, 8, 130, 44];

const found = array1.find(element => element > 10);

console.log(found);
// expected output: 12

```

Array.pop()

The pop() method removes the last element from an array and returns that element. This method changes the length of the array.

JavaScript Demo: Array.pop()

```
const plants = ['broccoli', 'cauliflower', 'cabbage', 'kale', 'tomato'];

console.log(plants.pop());
// expected output: "tomato"

console.log(plants);
// expected output: Array ["broccoli", "cauliflower", "cabbage", "kale"]

plants.pop();

console.log(plants);
// expected output: Array ["broccoli", "cauliflower", "cabbage"]
```

Array.map()

The map() method creates a new array populated with the results of calling a provided function on every element in the calling array.

JavaScript Demo: Array.map()

```
const array1 = [1, 4, 9, 16];

// pass a function to map
const map1 = array1.map(x => x * 2);

console.log(map1);
// expected output: Array [2, 8, 18, 32]

```

Array.push()

The push() method adds one or more elements to the end of an array and returns the new length of the array.

JavaScript Demo: Array.push()

```
1  const animals = ['pigs', 'goats', 'sheep'];
2
3  const count = animals.push('cows');
4  console.log(count);
5  // expected output: 4
6  console.log(animals);
7  // expected output: Array ["pigs", "goats", "sheep", "cows"]
8
9  animals.push('chickens', 'cats', 'dogs');
10 console.log(animals);
11 // expected output: Array ["pigs", "goats", "sheep", "cows", "chickens", "cats", "dogs"]
12
```

Array.shift()

The shift() method removes the first element from an array and returns that removed element. This method changes the length of the array.

JavaScript Demo: Array.shift()

```
1  const array1 = [1, 2, 3];
2
3  const firstElement = array1.shift();
4
5  console.log(array1);
6  // expected output: Array [2, 3]
7
8  console.log(firstElement);
9  // expected output: 1
10
```

Array.unshift()

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

JavaScript Demo: Array.unshift()

```
const array1 = [1, 2, 3];

console.log(array1.unshift(4, 5));
// expected output: 5

console.log(array1);
// expected output: Array [4, 5, 1, 2, 3]

```

Array.includes()

The includes() method determines whether an array includes a certain value among its entries, returning true or false as appropriate.

JavaScript Demo: Array.includes()

```
const array1 = [1, 2, 3];

console.log(array1.includes(2));
// expected output: true

const pets = ['cat', 'dog', 'bat'];

console.log(pets.includes('cat'));
// expected output: true

console.log(pets.includes('at'));
// expected output: false
```

Array.splice()

The splice() method changes the contents of an array by removing or replacing existing elements and/or adding new elements in place.

JavaScript Demo: Array.splice()

```
const months = ['Jan', 'March', 'April', 'June'];
months.splice(1, 0, 'Feb');
// inserts at index 1
console.log(months);
// expected output: Array ["Jan", "Feb", "March", "April", "June"]

months.splice(4, 1, 'May');
// replaces 1 element at index 4
console.log(months);
// expected output: Array ["Jan", "Feb", "March", "April", "May"]

```

Array.slice()

The slice() method returns a shallow copy of a portion of an array into a new array object selected from start to end (end not included) where start and end represent the index of items in that array.

JavaScript Demo: Array.slice()

```
const animals = ['ant', 'bison', 'camel', 'duck', 'elephant'];

console.log(animals.slice(2));
// expected output: Array ["camel", "duck", "elephant"]

console.log(animals.slice(2, 4));
// expected output: Array ["camel", "duck"]

console.log(animals.slice(-2));
// expected output: Array ["duck", "elephant"]

console.log(animals.slice(2, -1));
// expected output: Array ["camel", "duck"]
```