

Arrays:

An array in JavaScript is a special type of variable that can hold multiple values. It is defined using square brackets [] and can contain elements of any data type (numbers, strings, objects, other arrays, etc.).

```
// Example of an array declaration
let numbers = [1, 2, 3, 4, 5];
let fruits = ['apple', 'banana', 'orange'];
let mixedArray = [1, 'hello', true, { name: 'John' }];
```

Operations on Arrays

Printing an array: Array elements will be printed.

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

Accessing Elements: Elements in an array are accessed using zero-based indexing.

```
let fruits = ['apple', 'banana', 'orange'];
console.log(fruits[0]); // Output: 'apple'
```

Modifying Elements: Elements in an array can be modified directly by accessing them via their index.

```
let fruits = ['apple', 'banana', 'orange'];
fruits[1] = 'grape';
console.log(fruits); // Output: ['apple', 'grape', 'orange']
```

Array Length: You can find the number of elements in an array using the length property.

```
let fruits = ['apple', 'banana', 'orange'];
console.log(fruits.length); // Output: 3
```

Adding Elements: You can add elements to the end of an array using the push() method.

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

Removing Elements: Elements can be removed from the end of an array using the `pop()` method.

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

Join Operation:

```
array.join(separator);
```

```
let fruits = ['apple', 'banana', 'orange'];
let result = fruits.join(); // default separator is ','
console.log(result); // Output: 'apple,banana,orange'
```

```
let fruits = ['apple', 'banana', 'orange'];
let result = fruits.join(' and ');
console.log(result); // Output: 'apple and banana and orange'
```

shift()

The `shift()` method removes the first element from an array and returns that removed element. This operation also shifts all subsequent elements to the left by one position.

Syntax

```
array.shift()
```

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

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

In this example, `shift()` removes 'apple' from the `fruits` array and returns it. The remaining elements ('banana' and 'orange') are shifted left by one position.

unshift()

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

```
array.unshift(element1, element2, ..., elementN)
```

array: The array to which elements are to be added.

element1, element2, ..., elementN: Elements to add to the beginning of the array.

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

```
let newLength = fruits.unshift('apple', 'grape');  
console.log(newLength); // Output: 4 (length of the modified array)  
console.log(fruits); // Output: ['apple', 'grape', 'banana', 'orange']
```

1. `slice()`

The `slice()` method returns a shallow copy of a portion of an array into a new array object. It takes two arguments: the starting index (inclusive) and the ending index (exclusive).

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

```
let slicedFruits = fruits.slice(1, 3);  
console.log(slicedFruits); // Output: ['banana', 'cherry']
```

2. `concat()`

The `concat()` method is used to merge two or more arrays. It does not change the existing arrays, but instead returns a new array.

```
let arr1 = [1, 2, 3];
```

```
let arr2 = [4, 5, 6];
```

```
let mergedArray = arr1.concat(arr2);
```

```
console.log(mergedArray); // Output: [1, 2, 3, 4, 5, 6]
```

Standard Problems:

1. Sum of Array Elements

```
function sumArray(arr) {  
    let sum = 0;  
    for (let i = 0; i < arr.length; i++) {  
        sum += arr[i];  
    }  
    return sum;  
}  
  
let numbers = [1, 2, 3, 4, 5];  
console.log(sumArray(numbers)); // Output: 15
```

2. Product of Array Elements

```
function productArray(arr) {  
    let product = 1;  
    for (let i = 0; i < arr.length; i++) {  
        product *= arr[i];  
    }  
    return product;  
}
```

```
}
```

```
let numbers = [1, 2, 3, 4, 5];
```

```
console.log(productArray(numbers)); // Output: 120
```

3. Reverse an Array

```
function reverseArray(arr) {
```

```
    let reversed = [];
```

```
    for (let i = arr.length - 1; i >= 0; i--) {
```

```
        reversed.push(arr[i]);
```

```
    }
```

```
    return reversed;
```

```
}
```

```
let numbers = [1, 2, 3, 4, 5];
```

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

OR

```
function reverseArray(arr) {
```

```
    let start = 0;
```

```
    let end = arr.length - 1;
```

```
while (start < end) {  
    // Swap elements at start and end indices  
  
    let temp = arr[start];  
    arr[start] = arr[end];  
    arr[end] = temp;  
  
    // Move indices towards the center  
  
    start++;  
    end--;  
}  
  
return arr;  
}  
  
// Example usage:  
  
let numbers = [1, 2, 3, 4, 5];  
  
console.log(reverseArray(numbers)); // Output: [5, 4, 3, 2, 1]
```

4. Find Maximum and Minimum in an Array

```
function findMax(arr) {  
    let max = arr[0];  
    for (let i = 1; i < arr.length; i++) {
```

```
        if (arr[i] > max) {  
            max = arr[i];  
        }  
    }  
    return max;  
}
```

```
function findMin(arr) {  
    let min = arr[0];  
    for (let i = 1; i < arr.length; i++) {  
        if (arr[i] < min) {  
            min = arr[i];  
        }  
    }  
    return min;  
}
```

```
let numbers = [3, 7, 2, 1, 9, 4, 5];  
console.log(findMax(numbers)); // Output: 9  
console.log(findMin(numbers)); // Output: 1
```

5. Prefix Sum and Suffix Sum

- **Prefix Sum:** Sum of all elements from the start of the array up to a specific index.

```
function prefixSum(arr) {  
    let prefixSums = [];  
    let sum = 0;  
    for (let i = 0; i < arr.length; i++) {  
        sum += arr[i];  
        prefixSums.push(sum);  
    }  
    return prefixSums;  
}
```

```
let numbers = [1, 2, 3, 4, 5];  
console.log(prefixSum(numbers)); // Output: [1, 3, 6, 10, 15]
```

- **Suffix Sum:** Sum of all elements from a specific index to the end of the array.

```
function suffixSum(arr) {  
    let suffixSums = [];  
    let sum = 0;  
    for (let i = arr.length - 1; i >= 0; i--) {  
        sum += arr[i];  
        suffixSums.unshift(sum); // Add sum to the front of the array  
    }  
    return suffixSums;  
}
```



```
    }  
    return suffixSums;  
}
```

```
let numbers = [1, 2, 3, 4, 5];  
console.log(suffixSum(numbers)); // Output: [15, 14, 12, 9, 5]
```

7. Two Sum Problem (Finding Indices)

```
function twoSum(nums, target) {  
    let result = [];  
  
    // Iterate through each element in the array  
    for (let i = 0; i < nums.length; i++) {  
        for (let j = i + 1; j < nums.length; j++) {  
            // Check if nums[i] + nums[j] equals target  
            if (nums[i] + nums[j] === target) {  
                result.push(nums[i], nums[j]);  
                return result;  
            }  
        }  
    }  
}
```

```
// Example usage:

let numbers = [2, 7, 11, 15];

let target = 9;


console.log(twoSum(numbers, target)); // Output: [2, 7]

}
```

8. Linear Search

```
function linearSearch(arr, target) {
    for (let i = 0; i < arr.length; i++) {
        if (arr[i] === target) {
            return i; // Return the index of the target element
        }
    }

    return -1; // Target not found
}


let numbers = [5, 3, 8, 1, 9, 4];

console.log(linearSearch(numbers, 8)); // Output: 2 (index of the
element 8)

console.log(linearSearch(numbers, 10)); // Output: -1 (not found)
```

Check if an array forms palindrome or not

```
function isPalindrome(arr) {  
    let n = arr.length;  
    for (let i = 0; i < n / 2; i++) {  
        if (arr[i] !== arr[n - 1 - i]) {  
            return false;  
        }  
    }  
    return true;  
}
```

// Example usage:

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

```
let array2 = [1, 2, 3, 4, 5];
```

```
console.log(isPalindrome(array1)); // Output: true
```

```
console.log(isPalindrome(array2)); // Output: false
```

Homework: To explore map, find, reduce, filter functions

