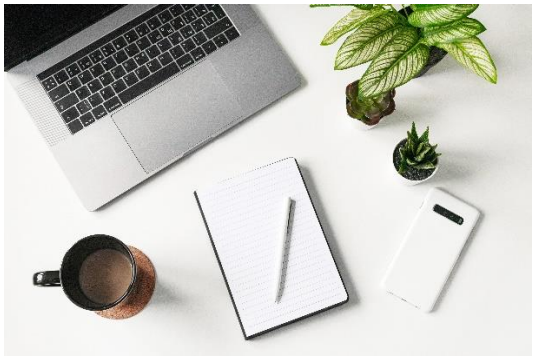

Most asked DSA question

Programming

Comprehensive Collection of Frequently Asked Data Structures and Algorithms (DSA) Questions for Freshers and Early Professionals

I have listed all the string methods, array methods and object method with example code and description.

SEPTEMBER 28, 2024

MUHAMMED BILAL

Q => add a word at the middle and at the end of a given string.

javascript Verify Open In Editor

```javascript
function addChar(str, word) {
  const words = " web";
  let res = "";
  return res = res + str.slice(0, 18) + words + str.slice(18) + word;
}

const str = "Javascript is best programming";
const word = " language";
console.log(addChar(str, word));
```

Q =>checking given 2 strings are angram or not

javascript Verify Open In Editor

```javascript
function isAnagram(str1, str2) {
  return str1.split("").sort().join("") === str2.split("").sort().join("");
}

const str1 = "baa";
const str2 = "aab";
console.log(isAngaram(str1, str2));
```

Q => Write a javascript program to find vowels counts

javascript Verify Open In Editor

```javascript
function countVowels(str) {
  const vowels = ["a", "e", "i", "o", "u", "A", "E", "I", "O", "U"];
  let count = 0;

  str.split("").forEach((char) => {
    if (vowels.includes(char)) {
      count++;
    }
  });

  return count;
}

const str = "javascript";
console.log(countVowels(str));
```

Q =>How to get the last character of a string in JavaScript

javascript Verify Open In Editor

```javascript
function lastCharOfStr(str){
    return str.slice(str.length-1)
}

const str='Javascript is best programming language'
console.log(lastCharOfStr(str))
```

Q => check given string is palindrome or not in JavaScript

javascript Verify Open In Editor

```javascript

function isPalindrome(str) {
  const reversedStr = str.split("").reverse().join("");
  return str === reversedStr;
}

const str = "java";
const str1 = "javascript";
console.log(isPalindrome(str)); // false
console.log(isPalindrome(str1)); // false
console.log(isPalindrome("madam")); // true
```

Q =>Delete first character of a string in JavaScript

javascript Verify Open In Editor

```javascript
function deleteFirstChar(str){
    //first method
    //let deletedChar = str.replace('J' , "")

    //second method
    let res = str.slice(1,str.length)
    return res
}


const str = 'Javascript';
console.log(deleteFirstChar(str))
```

Q =>How to remove text from a string in JavaScript

javascript Verify Open In Editor

```javascript
function removeWord(str, removingWord){
    //first method
    //let res = str.replace(removingWord,'')

    //second method
    const strArr = str.split(' ');
    let res = strArr.filter((item) => item !== removingWord)
    return res.join(' ')


}


const str='Javascript is best programming language'
const removingWord = 'best'
console.log(removeWord(str, removingWord))
```

Q =>How to count repeated char occurrence in string using JavaScript // 2 solutions

javascript Verify Open In Editor

```javascript
// Counts the repeated character occurrences in a given string.
function countRepeatedChar(str) {
  const charCountMap = new Map();
  for (const char of str) {
    charCountMap.set(char, (charCountMap.get(char) || 0) + 1);
  }
  return Array.from(charCountMap).filter(([key, value]) => value > 1);
}
const str = "Javascripteee";
console.log(countRepeatedChar(str));
```

javascript Verify Open In Editor

```javascript
function countRepeatedChar(str) {
  const myMap = new Map();
  for (const char of str) {
    if (myMap.has(char)) {
      myMap.set(char, myMap.get(char) + 1);
    } else {
      myMap.set(char, 1);
    }
  }
  const repeatedwords = [];
  for (const [key,value] of myMap) {
    if (myMap.get(key) > 1) {
            repeatedwords.push({ key, value });
    }
  }
  return repeatedwords;
}
const str = "Javascripteee";
console.log(countRepeatedChar(str));
```

Q =>Reverse a string in JavaScript

javascript Verify Open In Editor

```javascript
function reverseStr(str){
    // return str.split('').reverse().join('')
    let rev = '';
    for(let i=str.length-1;i>0; i--){
        rev= rev + str[i]
    }
    return rev
}
const str='Javascript is the best  programming  language'

console.log(reverseStr(str))
```

Q =>Sort a string in JavaScript

javascript Verify Open In Editor

```javascript
function sortString(str) {
  const strArr = str.split("").sort().join("");
  return strArr;
}

const str = "javascript";
console.log(sortString(str));
```

Q =>JavaScript program to check if a string contains given word

javascript Verify Open In Editor

```javascript
function containsWord(str, word) {
  return str.toLowerCase().includes(word.toLowerCase());
}

const str = "Javascript is best programming language";
const word = "best";
console.log(containsWord(str, word));
```

Q => How to make first letter of a string uppercase in JavaScript

javascript Verify Open In Editor

```javascript
function converToUpperCase(str) {
  //firstmethod
  // return str.toUpperCase()

  //second method
  let upperCase = "";
  for (const char of str) {
    upperCase = upperCase + char.toUpperCase();
  }
  return upperCase;
}

const str = "Javascript is best programming language";
console.log(converToUpperCase(str));
```

Q => Converts the first letter of a string to uppercase.

javascript Verify Open In Editor

```javascript

function capitalizeFirstLetter(str) {
  return str.charAt(0).toUpperCase() + str.slice(1);
}

const str = "javascript is best programming language";
console.log(capitalizeFirstLetter(str));
```

Q=> How to iterate over characters of a string in JavaScript

javascript Verify Open In Editor

```javascript
const str = "javascript";

for (let i = 0; i < str.length; i++) {
  console.log(str[i]);
}
```

javascript Verify Open In Editor

```javascript
const str = "javascript";

for (const char of str) {
  console.log(char);
}
```

Arrays Questions

Q => Reversing an Array in JavaScript

javascript Verify Open In Editor

```javascript
const arr = [1, 2, 3, 4, 5];
arr.reverse();
console.log(arr); // Output: [5, 4, 3, 2, 1]
```

javascript Verify Open In Editor

```javascript
const arr = [1, 2, 3, 4, 5];
const reversedArr = [];
for (let i = arr.length - 1; i ≥ 0; i--) {
  reversedArr.push(arr[i]);
}
console.log(reversedArr); // Output: [5, 4, 3, 2, 1]
```

Q => Find the maximum and minimum element in an array

javascript Verify Open In Editor

```javascript
const arr = [3, 1, 4, 10, 2, 6];
let max = -Infinity;
let min = Infinity;
for (let i = 0; i < arr.length; i++) {
  if (arr[i] > max) {
    max = arr[i];
  }
  if (arr[i] < min) {
    min = arr[i];
  }
}
console.log(`Maximum: ${max}, Minimum: ${min}`); // Output: Maximum: 10,
```

javascript Verify Open In Editor

```javascript
const arr = [3, 1, 4, 10, 2, 6];
let max = -Infinity;
let min = Infinity;

for (const element of arr) {
  if (element > max) {
    max = element;
  }
  if (element < min) {
    min = element;
  }
}

console.log(`Maximum: ${max}, Minimum: ${min}`); // Output: Maximum: 10, M
```

Q => Find the first duplicate element in an array

Using set

javascript Verify Open In Editor

```javascript
const arr = [2, 1, 3, 5, 3, 2];
function findFirstDuplicate(arr) {
  const seen = new Set();
  for (const element of arr) {
    if (seen.has(element)) {
      return element;
    }
    seen.add(element);
  }
  return null;
}
console.log(findFirstDuplicate(arr)); // Output: 3
```

Using Object

javascript Verify Open In Editor

```javascript
const arr = [2, 1, 3, 5, 3, 2];
function findFirstDuplicate(arr) {
  for (let i = 0; i < arr.length; i++) {
    for (let j = i + 1; j < arr.length; j++) {
      if (arr[i] === arr[j]) {
        return arr[i];
      }
    }
  }
  return null;
}
console.log(findFirstDuplicate(arr)); // Output: 3
```

javascript Verify Open In Editor

```javascript
const arr = [2, 1, 3, 5, 3, 2];
function findFirstDuplicate(arr) {
  const seen = {};
  for (const element of arr) {
    if (seen[element]) {
      return element;
    }
    seen[element] = true;
  }
  return null;
}
console.log(findFirstDuplicate(arr)); // Output: 3
```

Q => Find the missing element in an array

javascript Verify Open In Editor

```javascript
const arr1 = [1, 2, 3, 4, 5];
const arr2 = [1, 2, 3, 5];
function findMissingElement(arr1, arr2) {
  for (let i = 0; i < arr1.length; i++) {
    if (!arr2.includes(arr1[i])) {
      return arr1[i];
    }
  }
  return null;
}
console.log(findMissingElement(arr1, arr2)); // Output: 4
```

javascript Verify Open In Editor

```javascript
const arr1 = [1, 2, 3, 4, 5];
const arr2 = [1, 2, 3, 5];
function findMissingElement(arr1, arr2) {
  const presentInBoth = arr1.filter(element => arr2.includes(element));
  return arr1.find(element => !presentInBoth.includes(element));
}
console.log(findMissingElement(arr1, arr2)); // Output: 4
```

Q => Check if a Element is Present in an Array Using JavaScript

javascript Verify Open In Editor

```javascript
const arr = [1, 2, 3, 4, 5];
const element = 3;
console.log(arr.includes(element)); // Output: true
```

javascript Verify Open In Editor

```javascript
function checkElementPresent(arr, ele) {
  for (const item of arr) {
    if (item === ele) {
      return true;
    }
  }
  return false;
}
let arr = [19, 11, 15, 17, 12, 17, 14, 101, 23, 55, 77, 100];
const ele = 14
console.log(checkElementPresent(arr, ele)) // Output: true
```

Q => Sort an array

javascript Verify Open In Editor

```javascript
// Quicksort Algorithm Implementation
function quicksort(arr) {
array
  if (arr.length ≤ 1) {
    return arr;
  }
  let pivot = arr[0];
  const less = [];
  const greater = [];

  for (let i = 1; i < arr.length; i++) {
    // Check if the current element is less than or equal to the pivot
    if (arr[i] ≤ pivot) {
      // Add the element to the less array
      less.push(arr[i]);
    } else {
      // Add the element to the greater array
      greater.push(arr[i]);
    }
  }
  return quicksort(less).concat(pivot, quicksort(greater));
}

let arr = [19, 11, 15, 17, 12, 17, 14, 101, 23, 55, 77, 100];
console.log(quicksort(arr));
```

javascript Verify Open In Editor

```javascript
const arr = [5, 2, 8, 3, 1, 6, 4];
arr.sort((a, b) => a - b);
console.log(arr); // Output: [1, 2, 3, 4, 5, 6, 8]
```

Q => program to check if two arrays are equal or not

javascript Verify Open In Editor

```javascript
function areArraysEqual(arr1, arr2) {
  return JSON.stringify(arr1) === JSON.stringify(arr2);
}
```

This method converts the arrays to JSON strings and compares the strings. If the strings are equal, the arrays are equal.

Another way

javascript Verify Open In Editor

```javascript
function areArraysEqual(arr1, arr2) {
  // Check if the lengths of the two arrays are equal
  if (arr1.length !== arr2.length) {
    return false;
  }

  for (let i = 0; i < arr1.length; i++) {
    // Check if the elements at the current index are equal
    if (arr1[i] !== arr2[i]) {
      return false;
    }
  }
  return true;
}

let arr1 = [1, 2, 3, 4, 5];
let arr2 = [1, 2, 3, 4, 5];
console.log(areArraysEqual(arr1, arr2)); // Output: true
```

Q => write a function to calculates the factorial of a number

javascript Verify Open In Editor

```javascript
function factorial(n) {
  if (n === 0 || n === 1) {
    return 1;
  } else {
    return n * factorial(n - 1);
  }
}

console.log(factorial(5)); // Output: 120

```

javascript Verify Open In Editor

```javascript
function factorial(n) {
  let result = 1;
  for (let i = 1; i <= n; i++) {
    result *= i;
  }
  return result;
}

console.log(factorial(5)); // Output: 120
```

Q => program that generates the Fibonacci

javascript Verify Open In Editor

```javascript
function fibonacci(n) {
  let fib = [0, 1];
  for (let i = 2; i < n; i++) {
    fib[i] = fib[i - 1] + fib[i - 2];
  }
  return fib;
}

console.log(fibonacci(10));
```

javascript Verify Open In Editor

```javascript
function fibonacci(n) {
  let fib = [0, 1];
  for (let i = 2; i < n; i++) {
    fib[i] = fib[i - 1] + fib[i - 2];
  }
  return fib[n - 1];
}

console.log(fibonacci(10));
```

Q => function that finds duplicates in an array:

javascript Verify Open In Editor

```javascript
// Function to find duplicates in an array
function findDuplicates(arr) {
  let duplicates = [];
  for (let i = 0; i < arr.length; i++) {
    if (arr.indexOf(arr[i]) !== arr.lastIndexOf(arr[i]) && !duplicates.i
      duplicates.push(arr[i]);
    }
  }
  return duplicates;
}

// Test the function
let arr = [1, 2, 3, 2, 4, 5, 5, 6, 7, 8, 8, 9];
console.log(findDuplicates(arr)); // Output: [2, 5, 8]
```

Using Map

javascript Verify Open In Editor

```javascript
// Function to find duplicates in an array
function findDuplicates(arr) {
  let countMap = new Map();
  let duplicates = [];
  for (let num of arr) {
    if (countMap.has(num)) {
      countMap.set(num, countMap.get(num) + 1);
    } else {
      countMap.set(num, 1);
    }
  }
  for (let [num, count] of countMap) {
    if (count > 1) {
      duplicates.push(num);
    }
  }
  return duplicates;
}

// Test the function
let arr = [1, 2, 3, 2, 4, 5, 5, 6, 7, 8, 8, 9];
console.log(findDuplicates(arr)); // Output: [2, 5, 8]
```

Q => function that flattens a nested array:

javascript Verify Open In Editor

```javascript
// Function to flatten a nested array
function flattenArray(arr) {
  let flatArr = [];
  for (let item of arr) {
    if (Array.isArray(item)) {
      flatArr = flatArr.concat(flattenArray(item));
    } else {
      flatArr.push(item);
    }
  }
  return flatArr;
}

// Test the function
let arr = [1, 2, [3, 4, [5, 6]], 7, [8, 9]];
console.log(flattenArray(arr)); // Output: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Using in-build method

javascript Verify Open In Editor

```javascript
// Function to flatten a nested array
function flattenArray(arr) {
  return arr.flat(Infinity);
}

// Test the function
let arr = [1, 2, [3, 4, [5, 6]], 7, [8, 9]];
console.log(flattenArray(arr)); // Output: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Q => function that finds the largest element in an array:

javascript Verify Open In Editor

```javascript
// Function to find the largest element in an array
function findLargest(arr) {
  let largest = arr[0];
  for (let i = 1; i < arr.length; i++) {
    if (arr[i] > largest) {
      largest = arr[i];
    }
  }
  return largest;
}

// Test the function
let arr = [1, 2, 3, 4, 5, 6, 7, 8, 9];
console.log(findLargest(arr)); // Output: 9
```

javascript Verify Open In Editor

```javascript
 function findLargest(arr) {
  return Math.max(...arr);
}
 let arr = [1, 2, 3, 4, 5, 6, 7, 8, 9];
console.log(findLargest(arr)); // Output: 9
```

Q => Find largest sum of two numbers

javascript Verify Open In Editor

```javascript
// Function to find the largest sum of pairs in an array
function findLargestSumOfPairs(arr) {
  let largestSum = -Infinity;
  for (let i = 0; i < arr.length; i++) {
    for (let j = i + 1; j < arr.length; j++) {
      let sum = arr[i] + arr[j];
      if (sum > largestSum) {
        largestSum = sum;
      }
    }
  }
  return largestSum;
}

// Test the function
let arr = [1, 2, 3, 4, 5, 6, 7, 8, 9];
console.log(findLargestSumOfPairs(arr)); // Output: 17
```

javascript Verify Open In Editor

```javascript
// Function to find the largest sum of pairs in an array
function findLargestSumOfPairs(arr) {
  arr.sort((a, b) => b - a);
  return arr[0] + arr[1];
}

// Test the function
let arr = [1, 2, 3, 4, 5, 6, 7, 8, 9];
console.log(findLargestSumOfPairs(arr)); // Output: 17
```

Q => function that calculates the maximum possible profit from buying and selling a stock:

javascript Verify Open In Editor

```javascript
// Function to calculate the maximum possible profit from buying and se
function maxProfit(prices) {
  let minPrice = prices[0];
  let maxProfit = 0;
  for (let i = 1; i < prices.length; i++) {
    if (prices[i] < minPrice) {
      minPrice = prices[i];
    } else if (prices[i] - minPrice > maxProfit) {
      maxProfit = prices[i] - minPrice;
    }
  }
  return maxProfit;
}

// Test the function
let prices = [7, 1, 5, 3, 6, 4];
console.log(maxProfit(prices)); // Output: 5
```

javascript Verify Open In Editor

```javascript
// Function to calculate the maximum possible profit from buying and sel
function maxProfit(prices) {
  let minPrice = Infinity;
  let maxProfit = 0;
  for (let price of prices) {
    minPrice = Math.min(minPrice, price);
    maxProfit = Math.max(maxProfit, price - minPrice);
  }
  return maxProfit;
}

// Test the function
let prices = [7, 1, 5, 3, 6, 4];
console.log(maxProfit(prices)); // Output: 5
```

Q => JavaScript program that solves the Two Sum problem:

javascript Verify Open In Editor

```javascript
// Function to find two numbers in an array that add up to a given targe
function twoSum(nums, target) {
  let numMap = new Map();
  for (let i = 0; i < nums.length; i++) {
    let complement = target - nums[i];
    if (numMap.has(complement)) {
      return [numMap.get(complement), i];
    }
    numMap.set(nums[i], i);
  }
  return null;
}

// Test the function
let nums = [2, 7, 11, 15];
let target = 9;
console.log(twoSum(nums, target)); // Output: [0, 1]
```

Q => function that finds the second largest element in an array:

javascript Verify Open In Editor

```javascript
// Function to find the second largest element in an array
function secondLargest(arr) {
  if (arr.length < 2) {
    throw new Error("Array must have at least two elements");
  }
  const sortedArr = arr.sort((a, b) => a - b);
  return sortedArr[arr.length - 2];
}

const arr = [1, 3, 44, 5, 6, 77, 88, 4, 22, 111];
console.log(secondLargest(arr));
```

Q => Function to modify the employee data

javascript Verify Open In Editor

```javascript
// Function to modify the employee data
function modifiedEmployees(arr) {
  const changeName = "Moin";
  const modified = arr.map((item) => {
    if (item.employee_name === "Rahul") {
      return { ...item, employee_name: changeName };
    }
    return item;
  });
  return modified;
}

// Test the function
let employees_data = [
  {
    employee_id: 1,
    employee_name: "Bilal",
  },
  {
    employee_id: 2,
    employee_name: "Rahul",
  },
  {
    employee_id: 3,
    employee_name: "Sameer",
  },
];

console.log(modifiedEmployees(employees_data));
```

string methods in JavaScript, along with example code:

1. charAt()

Returns the character at the specified index.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.charAt(0)); // Output: "H"
```

2. charCodeAt()

Returns the Unicode value of the character at the specified index.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.charCodeAt(0)); // Output: 72
```

3. concat()

Concatenates two or more strings.

javascript Verify Open In Editor

```javascript
let str1 = "Hello";
let str2 = " World";
console.log(str1.concat(str2)); // Output: "Hello World"
```

4. endsWith()

Returns true if the string ends with the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.endsWith("World")); // Output: true
```

5. fromCharCode()

Returns a string created from the specified sequence of Unicode values.

javascript Verify Open In Editor

```javascript
console.log(String.fromCharCode(72, 101, 108, 108, 111)); // Output: "Hello"
```

6. includes()

Returns true if the string contains the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.includes("World")); // Output: true
```

7. indexOf()

Returns the index of the first occurrence of the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.indexOf("World")); // Output: 6
```

8. lastIndexOf()

Returns the index of the last occurrence of the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.lastIndexOf("World")); // Output: 6
```

9. localeCompare()

Compares two strings in a locale-specific manner.

javascript Verify Open In Editor

```javascript
let str1 = "apple";
let str2 = "banana";
console.log(str1.localeCompare(str2)); // Output: -1
```

10. match()

Returns an array of matches for the specified regular expression.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.match(/World/)); // Output: ["World"]
```

12. padEnd()

Pads the string with the specified value to the specified length.

javascript Verify Open In Editor

```javascript
let str = "Hello";
console.log(str.padEnd(10, " World")); // Output: "Hello World"
```

13. padStart()

Pads the string with the specified value to the specified length.

javascript Verify Open In Editor

```javascript
let str = "Hello";
console.log(str.padStart(10, " World")); // Output: " WorldHello"
```

14. repeat()

Returns a string repeated the specified number of times.

javascript Verify Open In Editor

```javascript
let str = "Hello";
console.log(str.repeat(3)); // Output: "HelloHelloHello"
```

15. replace()

Replaces the specified value with the specified replacement.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.replace("World", "Universe")); // Output: "Hello Universe"
```

16. search()

Returns the index of the first occurrence of the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.search("World")); // Output: 6
```

17. slice()

Returns a substring of the specified length.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.slice(6, 11)); // Output: "World"
```

18. split()

Splits the string into an array of substrings.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.split(" ")); // Output: ["Hello", "World"]
```

19. startsWith()

Returns true if the string starts with the specified value.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.startsWith("Hello")); // Output: true
```

20. substring()

Returns a substring of the specified length.

javascript Verify Open In Editor

```javascript
let str = "Hello World";
console.log(str.substring(6, 11)); // Output: "World"
```

21. toLocaleLowerCase()

Returns the string in lowercase, using the locale-specific case mapping.

javascript Verify Open In Editor

```javascript
let str = "HELLO";
console.log(str.toLocaleLowerCase()); // Output: "hello"
```

22. toLocaleUpperCase()

Returns the string in uppercase, using the locale-specific case mapping.

javascript Verify Open In Editor

```javascript
let str = "hello";
console.log(str.toLocaleUpperCase()); // Output: "HELLO"
```

23. toLowerCase()

Returns the string in lowercase.

javascript Verify Open In Editor

```javascript
let str = "HELLO";
console.log(str.toLowerCase()); // Output: "hello
```

all the array methods in JavaScript, along with example code:

1. concat()

Returns a new array that is the result of concatenating two or more arrays.

javascript Verify Open In Editor

```javascript
let arr1 = [1, 2, 3];
let arr2 = [4, 5, 6];
console.log(arr1.concat(arr2)); // Output: [1, 2, 3, 4, 5, 6]
```

2. copyWithin()

Copies a sequence of elements within the array to a new position.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
arr.copyWithin(0, 2);
console.log(arr); // Output: [3, 4, 5, 4, 5]
```

3. entries()

Returns an iterator object that contains the key-value pairs of the array.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3];
for (let [index, value] of arr.entries()) {
  console.log(`Index: ${index}, Value: ${value}`);
}
// Output:
// Index: 0, Value: 1
// Index: 1, Value: 2
// Index: 2, Value: 3
```

4. every()

Returns true if all elements in the array pass the test implemented by the provided function.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.every(x => x > 0)); // Output: true
```

5. fill()

Fills all the elements of an array from a start index to an end index with a static value.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
arr.fill(0, 2, 4);
console.log(arr); // Output: [1, 2, 0, 0, 5]
```

6. filter()

Creates a new array with all elements that pass the test implemented by the provided function.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.filter(x => x > 3)); // Output: [4, 5]
```

7. find()

Returns the first element in the array that satisfies the provided testing function.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.find(x => x > 3)); // Output: 4
```

8. findIndex()

Returns the index of the first element in the array that satisfies the provided testing function.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.findIndex(x => x > 3)); // Output: 3
```

9. forEach()

Executes a provided function once for each array element.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
arr.forEach(x => console.log(x));
// Output:
// 1
// 2
// 3
// 4
// 5
```

10. includes()

Determines whether an array includes a certain element, returning true or false as appropriate.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.includes(3)); // Output: true
```

11. indexOf()

Returns the first index at which a given element can be found in the array, or -1 if it is not present.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.indexOf(3)); // Output: 2
```

12. join()

Joins all elements of an array into a string.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.join(", ")); // Output: "1, 2, 3, 4, 5"
```

13. keys()

Returns a new Array Iterator object that contains the keys for each index in the array.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3];
for (let key of arr.keys()) {
  console.log(key);
}
// Output:
// 0
// 1
// 2
```

14. lastIndexOf()

Returns the last index at which a given element can be found in the array, or -1 if it is not present.

javascript Verify Open In Editor

```javascript
let arr = [1, 2, 3, 4, 5];
console.log(arr.lastIndexOf(3)); // Output: 2
```

all the methods for objects in JavaScript, along with example code:

1. Object.assign()

Copies the values of all enumerable own properties from one or more source objects to a target object.

javascript Verify Open In Editor

```javascript
let target = { a: 1 };
let source = { b: 2 };
Object.assign(target, source);
console.log(target); // Output: { a: 1, b: 2 }
```

2. Object.create()

Creates a new object, using an existing object as the prototype of the newly created object.

javascript Verify Open In Editor

```javascript
let proto = { a: 1 };
let obj = Object.create(proto);
console.log(obj.a); // Output: 1
```

3. Object.defineProperty()

Defines a new property directly on an object, or modifies an existing property on an object, and returns the object.

javascript Verify Open In Editor

```javascript
let obj = {};
Object.defineProperty(obj, "a", { value: 1 });
console.log(obj.a); // Output: 1
```

4. Object.defineProperties()

Defines multiple new properties directly on an object, or modifies existing properties on an object, and returns the object.

javascript Verify Open In Editor

```javascript
let obj = {};
Object.defineProperties(obj, {
  a: { value: 1 },
  b: { value: 2 }
});
console.log(obj.a); // Output: 1
console.log(obj.b); // Output: 2
```

5. Object.entries()

Returns an array of a given object's own enumerable string-keyed property [key, value] pairs.

javascript Verify Open In Editor

```javascript
let obj = { a: 1, b: 2 };
console.log(Object.entries(obj)); // Output: [["a", 1], ["b", 2]]
```

6. Object.freeze()

Freezes an object: that is, prevents new properties from being added to it; prevents existing properties from being removed; and prevents existing properties, or their enumerability, configurability, or writability, from being changed.

javascript Verify Open In Editor

```javascript
let obj = { a: 1 };
Object.freeze(obj);
obj.a = 2; // throws a TypeError in strict mode
```

7. Object.fromEntries()

Transforms a list of key-value pairs into an object.

javascript Verify Open In Editor

```javascript
let entries = [["a", 1], ["b", 2]];
console.log(Object.fromEntries(entries)); // Output: { a: 1, b: 2 }
```

8. Object.getOwnPropertyDescriptor()

Returns a property descriptor for an own property (that is, one directly present on an object, not present by dint of being along an object's prototype chain) of a given object.

javascript Verify Open In Editor

```javascript
let obj = { a: 1 };
console.log(Object.getOwnPropertyDescriptor(obj, "a")); // Output: { value
```

13. Object.hasOwn()

Returns a boolean indicating whether the given object owns the specified property as its own property (as opposed to inheriting it).

javascript Verify Open In Editor

```javascript
let obj = { a: 1 };
console.log(Object.hasOwn(obj, "a")); // Output: true
```

14. Object.is()

Determines whether two values are the same value.

javascript Verify Open In Editor

```javascript
console.log(Object.is(1, 1)); // Output: true
console.log(Object.is(1, "1")); // Output: false
```

15. Object.isExtensible()

Determines whether an object is extensible.

javascript Verify Open In Editor

```javascript
let obj = { a: 1 };
console.log(Object.isExtensible(obj)); // Output:
```