JavaScript basic

1. Display current day and time

```
function displayDateTime() {
    let now = new Date();

    // Array of day names
    let days = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday',
'Friday', 'Saturday'];

    // Get day of the week
    let day = days[now.getDay()];

    // Get current time
    let hours = now.getHours();
    let minutes = now.getMinutes();
    let seconds = now.getSeconds();
    let period = hours >= 12 ? 'PM' : 'AM';

    // Convert to 12-hour format
    hours = hours % 12;
    hours = hours ? hours : 12; // Handle midnight (0 hours)

    // Displaying the output
    console.log(`Today is : ${day}.`);
    console.log(`Current time is : ${hours} ${period} : ${minutes} :
${seconds}`);
}

// Calling the function to display current day and time
displayDateTime();
```

2. Print the contents of the current window

```
function printCurrentWindow() {
    window.print();
}

// Call the function to print current window content
printCurrentWindow();
```

3. Get the current date in different formats

```
function getCurrentDate(format) {
    let now = new Date();
    let day = now.getDate();
    let month = now.getMonth() + 1; // Months are zero-indexed
    let year = now.getFullYear();

    switch (format) {
        case 'mm-dd-yyyy':
            return `${month}-${day}-${year}`;
        case 'mm/dd/yyyy':
            return `${month}/${day}/${year}`;
        case 'dd-mm-yyyy':
            return `${day}-${month}-${year}`;
        case 'dd/mm/yyyy':
            return `${day}/${month}/${year}`;
        default:
            return `${month}-${day}-${year}`; // Default format
    }
}

// Example usage
console.log(getCurrentDate('mm-dd-yyyy')); // Change format as needed
```

4. Calculate the area of a triangle given sides 5, 6, 7

```
function calculateTriangleArea(a, b, c) {
    // Using Heron's formula to calculate area
    let s = (a + b + c) / 2;
    let area = Math.sqrt(s * (s - a) * (s - b) * (s - c));
    return area;
}

// Given sides
let side1 = 5;
let side2 = 6;
let side3 = 7;

// Calculate and display area
console.log(`Area of the triangle is: ${calculateTriangleArea(side1, side2,
side3)}`);
```

5. Rotate the string 'w3resource' in right direction

```
function rotateStringRight(str) {
    let result = str[str.length - 1] + str.substring(0, str.length - 1);
    return result;
}

// Given string
let str = 'w3resource';

// Rotate and display
console.log(rotateStringRight(str));
```

6. Determine if a given year is a leap year

```
function isLeapYear(year) {
    return (year % 4 === 0 && year % 100 !== 0) || (year % 400 === 0);
}

// Example usage
let year = 2024;
```

```
console.log(`Is ${year} a leap year? ${isLeapYear(year)}`);
```

7. Find 1st January that is a Sunday between 2014 and 2050

```
function findFirstJanSunday() {
    for (let year = 2014; year <= 2050; year++) {
        let date = new Date(year, 0, 1); // January 1st
        if (date.getDay() === 0) { // 0 corresponds to Sunday
            return date.getFullYear();
        }
    }
    return null; // If no such year is found
}

// Call the function
console.log(`1st January is a Sunday in the year: ${findFirstJanSunday()}`);
```

8. Guessing game with random number

```
function guessingGame() {
    // Generate a random number between 1 and 10
    let randomNumber = Math.floor(Math.random() * 10) + 1;

    // Prompt user for input
    let guess = prompt('Guess the number between 1 and 10:');

    // Convert guess to number (since prompt returns a string)
    guess = parseInt(guess);

    // Check if guess matches randomNumber
    if (guess === randomNumber) {
        alert('Good Work!');
    } else {
        alert('Not matched. The number was ' + randomNumber);
    }
}

// Call the function to start the game
guessingGame();
```

9. Calculate days left until next Christmas

```
function daysUntilChristmas() {
    let today = new Date();
    let christmas = new Date(today.getFullYear(), 11, 25); // 11 is December
(zero-indexed month)

    if (today.getMonth() === 11 && today.getDate() > 25) {
        christmas.setFullYear(christmas.getFullYear() + 1);
    }

    // Calculate the difference in milliseconds
    let oneDay = 1000 * 60 * 60 * 24;
    let daysLeft = Math.ceil((christmas - today) / oneDay);
    return daysLeft;
}

// Call the function and display days left
console.log(`Days left until Christmas: ${daysUntilChristmas()}`);
```

10. Calculate multiplication and division of two numbers

```
function performOperations() {
    let num1 = parseFloat(prompt('Enter the first number:'));
    let num2 = parseFloat(prompt('Enter the second number:'));

    // Perform operations
    let multiplication = num1 * num2;
    let division = num1 / num2;

    // Display results
    console.log(`Multiplication: ${num1} * ${num2} = ${multiplication}`);
    console.log(`Division: ${num1} / ${num2} = ${division}`);
}

// Call the function to perform operations
performOperations();
```

11. Convert temperatures between Celsius and Fahrenheit

```
function convertTemperature(temp, fromUnit) {
    if (fromUnit === 'C') {
        // Celsius to Fahrenheit
        let fahrenheit = (temp * 9/5) + 32;
        return `${temp}°C is ${fahrenheit}°F`;
    } else if (fromUnit === 'F') {
        // Fahrenheit to Celsius
        let celsius = (temp - 32) * 5/9;
        return `${temp}°F is ${celsius}°C`;
    } else {
        return 'Invalid input';
    }
}

// Example conversions
console.log(convertTemperature(60, 'C'));
console.log(convertTemperature(45, 'F'));
```

12. Get the current website URL

```
function getCurrentURL() {
    return window.location.href;
}

// Display current URL
console.log(`Current URL: ${getCurrentURL()}`);
```

JavaScript Function

1. **Reverse a Number:**

```
function reverseNumber(num) {
    let reversedNum = 0;
    while (num !== 0) {
        reversedNum = reversedNum * 10 + num % 10;
        num = Math.floor(num / 10);
    }
    return reversedNum;
}

// Example usage:
let x = 32243;
console.log(reverseNumber(x));  // Output: 34223
```

2. **Check Palindrome:**

```
function isPalindrome(str) {
    // Remove non-alphanumeric characters and convert to lowercase
    str = str.replace(/[^a-zA-Z0-9]/g, '').toLowerCase();
    const len = str.length;
    for (let i = 0; i < len / 2; i++) {
        if (str[i] !== str[len - 1 - i]) {
            return false;
        }
    }
    return true;
}

// Example usage:
let str1 = "madam";
let str2 = "nurses run";
console.log(isPalindrome(str1));  // Output: true
console.log(isPalindrome(str2));  // Output: true
```

3. **Generate All Combinations of a String:**

```
function generateCombinations(str) {
    const result = [];
    const len = str.length;
    for (let i = 0; i < len; i++) {
        for (let j = i + 1; j <= len; j++) {
            result.push(str.slice(i, j));
        }
    }
    return result;
}
```

```
// Example usage:
let string = 'dog';
console.log(generateCombinations(string));  // Output: ["d", "do", "dog",
"o", "og", "g"]
```

4. **Sort Letters in Alphabetical Order:**

```
function sortLetters(str) {
    return str.split('').sort().join('');
}

// Example usage:
let exampleString = 'webmaster';
console.log(sortLetters(exampleString));  // Output: "abeemrstw"
```

5. **Convert First Letter of Each Word to Uppercase:**

```
function capitalizeWords(str) {
    return str.replace(/\b\w/g, char => char.toUpperCase());
}

// Example usage:
let sentence = 'the quick brown fox';
console.log(capitalizeWords(sentence));  // Output: "The Quick Brown Fox"
```

6. **Find Longest Word in a String:**

```
function longestWord(str) {
    let words = str.split(' ');
    let maxLength = 0;
    let result = '';
    for (let word of words) {
        if (word.length > maxLength) {
            maxLength = word.length;
            result = word;
        }
    }
    return result;
}

// Example usage:
let text = 'Web Development Tutorial';
console.log(longestWord(text));  // Output: "Development"
```

7. **Count Vowels in a String:**

```
function countVowels(str) {
    const vowels = 'aeiouAEIOU';
    let count = 0;
    for (let char of str) {
        if (vowels.includes(char)) {
            count++;
        }
    }
    return count;
}

// Example usage:
let sentence = 'The quick brown fox';
console.log(countVowels(sentence));  // Output: 5
```

8. **Check if a Number is Prime:**

```
function isPrime(num) {
    if (num <= 1) {
        return false;
    }
    for (let i = 2; i <= Math.sqrt(num); i++) {
        if (num % i === 0) {
            return false;
        }
    }
    return true;
}

// Example usage:
let number = 17;
console.log(isPrime(number));  // Output: true
```

9. **Get Type of Argument:**

```
function getType(arg) {
    return typeof arg;
}

// Example usage:
console.log(getType({}));       // Output: "object"
console.log(getType(true));     // Output: "boolean"
console.log(getType(() => {})); // Output: "function"
console.log(getType(123));      // Output: "number"
console.log(getType("hello"));  // Output: "string"
console.log(getType(undefined));// Output: "undefined"
```

10. **Generate Identity Matrix:**

```
function generateIdentityMatrix(n) {
    let matrix = [];
    for (let i = 0; i < n; i++) {
        matrix[i] = [];
        for (let j = 0; j < n; j++) {
            matrix[i][j] = (i === j) ? 1 : 0;
        }
    }
    return matrix;
}

// Example usage:
let n = 3;
console.log(generateIdentityMatrix(n));
// Output: [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

11. **Find Second Lowest and Second Greatest Numbers:**

```
function findSecondLowestAndGreatest(arr) {
    if (arr.length < 2) {
        return "Array length must be at least 2";
    }

    // Sort the array
    arr.sort(function(a, b) { return a - b; });

    // Remove duplicates if any
    arr = arr.filter(function(value, index, self) {
        return self.indexOf(value) === index;
    });

    // If array has only unique elements, second lowest is at index 1 and
second greatest at the second last index.
    return [arr[1], arr[arr.length - 2]];
}

// Example usage:
let sampleArray = [1, 2, 3, 4, 5];
console.log(findSecondLowestAndGreatest(sampleArray));  // Output: [2, 4]
```

12. **Check if a Number is Perfect:**

```
function isPerfectNumber(num) {
    if (num <= 1) {
        return false;
    }

    let sum = 1; // Sum starts with 1 (since 1 is always a divisor)
    for (let i = 2; i <= Math.sqrt(num); i++) {
        if (num % i === 0) {
            sum += i;
            if (i !== num / i) {
                sum += num / i;
            }
        }
    }

    return sum === num;
}

// Example usage:
console.log(isPerfectNumber(6));   // Output: true
console.log(isPerfectNumber(28));  // Output: true
console.log(isPerfectNumber(496)); // Output: true
console.log(isPerfectNumber(8128));// Output: true
console.log(isPerfectNumber(10));  // Output: false
```

13. **Compute Factors of a Positive Integer:**

```
function computeFactors(num) {
    if (num <= 0 || !Number.isInteger(num)) {
        return "Please enter a positive integer.";
    }

    let factors = [];
    for (let i = 1; i <= Math.sqrt(num); i++) {
        if (num % i === 0) {
            factors.push(i);
            if (i !== num / i) {
                factors.push(num / i);
            }
        }
    }

    factors.sort(function(a, b) { return a - b; }); // Sorting factors
    return factors;
}

// Example usage:
console.log(computeFactors(28));  // Output: [1, 2, 4, 7, 14, 28]
```

14. **Convert Amount to Coins:**

```
function amountToCoins(amount, coins) {
    coins.sort(function(a, b) { return b - a; }); // Sort coins in descending
order
    let result = [];
    for (let i = 0; i < coins.length; i++) {
        while (amount >= coins[i]) {
            result.push(coins[i]);
            amount -= coins[i];
        }
    }
    return result;
}

// Example usage:
console.log(amountToCoins(46, [25, 10, 5, 2, 1]));  // Output: [25, 10, 10,
1]
```

15. **Compute the Value of bnb^nbn (Exponentiation):**

```
function power(base, exponent) {
    return Math.pow(base, exponent);
}

// Example usage:
```

```
let base = 2;
let exponent = 5;
console.log(power(base, exponent));  // Output: 32 (2^5)
```

16. **Extract Unique Characters from a String:**

```
function uniqueCharacters(str) {
    let unique = "";
    for (let char of str) {
        if (unique.indexOf(char) === -1) {
            unique += char;
        }
    }
    return unique;
}

// Example usage:
let exampleString = "thequickbrownfoxjumpsoverthelazydog";
console.log(uniqueCharacters(exampleString));  // Output:
"thequickbrownfxjmpsvlazydg"
```

17. **Count Occurrences of Each Letter in a String:**

```
function countLetters(str) {
    let result = {};
    str = str.toLowerCase().replace(/[^a-z]/g, ''); // Convert to lowercase
and remove non-alphabetic characters
    for (let char of str) {
        if (result[char]) {
            result[char]++;
        } else {
            result[char] = 1;
        }
    }
    return result;
}

// Example usage:
let sampleString = "The quick brown fox";
console.log(countLetters(sampleString));
// Output: { t: 1, h: 1, e: 1, q: 1, u: 1, i: 1, c: 1, k: 1, b: 1, r: 1, o:
2, w: 1, n: 1, f: 1, x: 1 }
```

18. **Binary Search in   Arrays:**

```
function binarySearch(arr, target) {
    let left = 0;
    let right = arr.length - 1;
    while (left <= right) {
        let mid = Math.floor((left + right) / 2);
        if (arr[mid] === target) {
            return mid;
        } else if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1; // Target not found
}

// Example usage:
let sortedArray = [1, 3, 5, 7, 9, 11, 13, 15];
console.log(binarySearch(sortedArray, 7));  // Output: 3 (index of 7)
```

19. **Return Array Elements Larger than a Number:**

```
function elementsLargerThan(arr, number) {
    return arr.filter(function(element) {
        return element > number;
    });
}

// Example usage:
let numbersArray = [2, 5, 8, 10, 15, 20];
console.log(elementsLargerThan(numbersArray, 8));  // Output: [10, 15, 20]
```

20. **Generate Random String ID of Specified Length:**

```
function generateRandomId(length) {
    const characters =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    let result = "";
    for (let i = 0; i < length; i++) {
        result += characters.charAt(Math.floor(Math.random() *
characters.length));
    }
    return result;
}

// Example usage:
console.log(generateRandomId(10));  // Output: Random string ID of length 10
```

21: Get all possible subsets with a fixed length

```
function getSubsets(arr, length) {
    let subsets = [];

    function backtrack(start, currentSubset) {
        if (currentSubset.length === length) {
            subsets.push(currentSubset.slice()); // Make a copy of
currentSubset
            return;
        }

        for (let i = start; i < arr.length; i++) {
            currentSubset.push(arr[i]);
            backtrack(i + 1, currentSubset);
            currentSubset.pop();
        }
    }

    backtrack(0, []);
    return subsets;
}

// Example usage:
const array = [1, 2, 3];
const subsetLength = 2;
console.log(getSubsets(array, subsetLength));
// Expected output: [[1, 2], [1, 3], [2, 3], [1, 2, 3]]
```

22: Count occurrences of a letter in a string

```
function countOccurrences(str, letter) {
    let count = 0;
    for (let i = 0; i < str.length; i++) {
        if (str.charAt(i) === letter) {
            count++;
        }
    }
    return count;
}

// Example usage:
const str = 'w3resource.com';
const letter = 'o';
console.log(countOccurrences(str, letter));
// Expected output: 2
```

23: Find the first not repeated character in a string

```
function firstNotRepeated(str) {
    let charCount = {};

    for (let char of str) {
        charCount[char] = (charCount[char] || 0) + 1;
    }

    for (let char of str) {
        if (charCount[char] === 1) {
            return char;
        }
    }

    return null; // If all characters repeat
}

// Example usage:
const inputStr = 'abacddbec';
console.log(firstNotRepeated(inputStr));
// Expected output: 'e'
```

24: Bubble Sort algorithm

```
function bubbleSort(arr) {
    let len = arr.length;
    for (let i = 0; i < len; i++) {
        for (let j = 0; j < len - 1 - i; j++) {
            if (arr[j] > arr[j + 1]) {
                // Swap elements
                let temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    return arr;
}

// Example usage:
const arrToSort = [12, 345, 4, 546, 122, 84, 98, 64, 9, 1, 3223, 455, 23,
234, 213];
console.log(bubbleSort(arrToSort));
// Expected output: [3223, 546, 455, 345, 234, 213, 122, 98, 84, 64, 23, 12,
9, 4, 1]
```

25: Find the longest country name in an array

```
function longestCountryName(countryNames) {
    let longestName = '';
    for (let country of countryNames) {
        if (country.length > longestName.length) {
            longestName = country;
        }
    }
    return longestName;
}

// Example usage:
const countries = ["Australia", "Germany", "United States of America"];
console.log(longestCountryName(countries));
// Expected output: "United States of America"
```

26: Find longest substring in a given string without repeating characters

```
function longestSubstringWithoutRepeating(str) {
    let longestSubstr = '';
    let currentSubstr = '';

    for (let char of str) {
        let index = currentSubstr.indexOf(char);
        if (index !== -1) {
            currentSubstr = currentSubstr.slice(index + 1);
        }
        currentSubstr += char;
        if (currentSubstr.length > longestSubstr.length) {
            longestSubstr = currentSubstr;
        }
    }

    return longestSubstr;
}

// Example usage:
const inputStr = 'abcabcbb';
console.log(longestSubstringWithoutRepeating(inputStr));
// Expected output: 'abc'
```

27: Find the longest palindrome in a given string

```
function longestPalindrome(str) {
    function expandAroundCenter(left, right) {
        while (left >= 0 && right < str.length && str[left] === str[right]) {
            left--;
            right++;
        }
        return str.slice(left + 1, right);
    }

    let longest = '';

    for (let i = 0; i < str.length; i++) {
        let oddPalindrome = expandAroundCenter(i, i);
        let evenPalindrome = expandAroundCenter(i, i + 1);

        if (oddPalindrome.length > longest.length) {
            longest = oddPalindrome;
        }
        if (evenPalindrome.length > longest.length) {
            longest = evenPalindrome;
        }
    }

    return longest;
}

// Example usage:
const inputStr = 'bananas';
console.log(longestPalindrome(inputStr));
// Expected output: 'anana'
```

28: Pass a   function as parameter

```
function higherOrderFunction(callback) {
    // Execute callback function
    callback();
}

function myCallbackFunction() {
    console.log('Callback function executed.');
}

// Example usage:
higherOrderFunction(myCallbackFunction);
// Output: 'Callback function executed.'
```

29: Get the function name

```
function getFunctionName(func) {
    return func.name;
}

// Example usage:
function exampleFunction() {
    // Function body
}

console.log(getFunctionName(exampleFunction));
// Expected output: 'exampleFunction'
```

JavaScript Recursion

1: Calculate the factorial of a number

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

// Example usage:
const number = 5;
console.log(`Factorial of ${number} is: ${factorial(number)}`);
// Expected output: Factorial of 5 is: 120
```

2: Find the greatest common divisor (gcd) of two positive numbers

```
function gcd(a, b) {
    if (b === 0) {
        return a;
    }
    return gcd(b, a % b);
}

// Example usage:
const num1 = 24;
const num2 = 36;
```

```
console.log(`GCD of ${num1} and ${num2} is: ${gcd(num1, num2)}`);
// Expected output: GCD of 24 and 36 is: 12
```

3: Get the integers in range (x, y)

```
function range(x, y) {
    let result = [];
    for (let i = x + 1; i < y; i++) {
        result.push(i);
    }
    return result;
}

// Example usage:
console.log(range(2, 9));
// Expected output: [3, 4, 5, 6, 7, 8]
```

4: Compute the sum of an array of integers

```
function sumArray(arr) {
    let sum = 0;
    for (let num of arr) {
        sum += num;
    }
    return sum;
}

// Example usage:
const array = [1, 2, 3, 4, 5, 6];
console.log(`Sum of array is: ${sumArray(array)}`);
// Expected output: Sum of array is: 21
```

5: Compute the exponent of a number

```
function exponent(base, exp) {
    return base ** exp;
}

// Example usage:
const base = 8;
const exponentValue = 2;
console.log(`${base}^${exponentValue} = ${exponent(base, exponentValue)}`);
// Expected output: 8^2 = 64
```

Problem 6: Get the first n Fibonacci numbers

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

```
// Example usage:
const count = 10;
console.log(`First ${count} Fibonacci numbers are: ${fibonacci(count)}`);
// Expected output: First 10 Fibonacci numbers are: [0, 1, 1, 2, 3, 5, 8, 13,
21, 34]
```

7: Check whether a number is even or not

```
function isEven(num) {
    return num % 2 === 0;
}

// Example usage:
const number = 6;
console.log(`${number} is even: ${isEven(number)}`);
// Expected output: 6 is even: true
```

8: Binary search in

```
Array.prototype.binarySearch = function(target) {
    let left = 0;
    let right = this.length - 1;

    while (left <= right) {
        let mid = Math.floor((left + right) / 2);
        if (this[mid] === target) {
            return mid;
        } else if (this[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1;
};

// Example usage:
const sortedArray = [0, 1, 2, 3, 4, 5, 6];
const target = 5;
console.log(`Index of ${target} in array is:
${sortedArray.binarySearch(target)}`);
// Expected output: Index of 5 in array is: 5
```

9: Merge sort in

```
function mergeSort(arr) {
    if (arr.length <= 1) {
        return arr;
    }

    const mid = Math.floor(arr.length / 2);
    const left = mergeSort(arr.slice(0, mid));
    const right = mergeSort(arr.slice(mid));
    return merge(left, right);
}

function merge(left, right) {
    let result = [];
    let leftIndex = 0;
    let rightIndex = 0;

    while (leftIndex < left.length && rightIndex < right.length) {
        if (left[leftIndex] < right[rightIndex]) {
            result.push(left[leftIndex]);
            leftIndex++;
        } else {
            result.push(right[rightIndex]);
            rightIndex++;
        }
    }

    return
result.concat(left.slice(leftIndex)).concat(right.slice(rightIndex));
}

// Example usage:
const arrayToSort = [34, 7, 23, 32, 5, 62];
console.log(`Sorted array: ${mergeSort(arrayToSort)}`);
// Expected output: Sorted array: [5, 7, 23, 32, 34, 62]
```

JavaScript conditional statements and loops

1: Display the larger of two integers

```
function displayLarger(num1, num2) {
    if (num1 > num2) {
        console.log(`${num1} is larger.`);
    } else if (num2 > num1) {
        console.log(`${num2} is larger.`);
    } else {
        console.log("Both numbers are equal.");
    }
}

// Example usage:
displayLarger(10, 5);
// Expected output: 10 is larger.
```

2: Find the sign of the product of three numbers

```
function findProductSign(num1, num2, num3) {
    let product = num1 * num2 * num3;
    let sign = product < 0 ? '-' : '+';
    alert(`The sign is ${sign}`);
}
```

```
// Example usage:
findProductSign(3, -7, 2);
// Expected alert: The sign is -
```

3: Sort three numbers and display in an alert box

```
function sortNumbers(num1, num2, num3) {
    let sortedArray = [num1, num2, num3].sort((a, b) => b - a);
    alert(`${sortedArray[0]}, ${sortedArray[1]}, ${sortedArray[2]}`);
}

// Example usage:
sortNumbers(0, -1, 4);
// Expected alert: 4, 0, -1
```

4: Find the largest of five numbers and display in an alert box

```
function findLargest(num1, num2, num3, num4, num5) {
    let largest = Math.max(num1, num2, num3, num4, num5);
    alert(`The largest number is: ${largest}`);
}

// Example usage:
findLargest(-5, -2, -6, 0, -1);
// Expected alert: The largest number is: 0
```

5: Check if numbers from 0 to 15 are odd or even

```
function checkOddEven() {
    for (let i = 0; i <= 15; i++) {
        if (i % 2 === 0) {
            console.log(`${i} is even`);
        } else {
            console.log(`${i} is odd`);
        }
    }
}

// Example usage:
checkOddEven();
// Expected output:
// "0 is even"
// "1 is odd"
// "2 is even"
// ...
// "15 is odd"
```

6: Compute average marks and determine grades

```
function computeGrade() {
    const students = [
        { name: 'David', marks: 80 },
        { name: 'Vinoth', marks: 77 },
        { name: 'Divya', marks: 88 },
        { name: 'Ishitha', marks: 95 },
        { name: 'Thomas', marks: 68 }
    ];

    let sum = 0;
    for (let student of students) {
        sum += student.marks;
    }

    let average = sum / students.length;

    let grade;
    if (average < 60) {
        grade = 'F';
    } else if (average < 70) {
        grade = 'D';
    } else if (average < 80) {
        grade = 'C';
    } else if (average < 90) {
        grade = 'B';
    } else {
        grade = 'A';
    }

    alert(`Average marks: ${average.toFixed(2)}\nGrade: ${grade}`);
}

// Example usage:
computeGrade();
// Example alert:
// Average marks: 81.60
// Grade: B
```

7: Print "Fizz", "Buzz", or "FizzBuzz" for multiples of 3, 5, or both

```
function fizzBuzz() {
    for (let i = 1; i <= 100; i++) {
        if (i % 3 === 0 && i % 5 === 0) {
            console.log("FizzBuzz");
        } else if (i % 3 === 0) {
            console.log("Fizz");
        } else if (i % 5 === 0) {
            console.log("Buzz");
        } else {
            console.log(i);
        }
    }
}

// Example usage:
fizzBuzz();
// Output:
// 1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, FizzBuzz,
...
```

8: Find the first 5 happy numbers

```
function isHappyNumber(num) {
    let seen = {};
    while (num !== 1 && !seen[num]) {
        seen[num] = true;
        num = sumOfSquaredDigits(num);
    }
    return num === 1;
}

function sumOfSquaredDigits(num) {
    let sum = 0;
    while (num > 0) {
        let digit = num % 10;
        sum += digit * digit;
        num = Math.floor(num / 10);
    }
    return sum;
}

function findHappyNumbers(count) {
    let happyNumbers = [];
    let num = 1;
    while (happyNumbers.length < count) {
        if (isHappyNumber(num)) {
            happyNumbers.push(num);
        }
        num++;
    }
    return happyNumbers;
}

// Example usage:
console.log(findHappyNumbers(5));
// Expected output: [1, 7, 10, 13, 19]
```

9: Find Armstrong numbers of 3 digits

```
function isArmstrong(num) {
    let strNum = num.toString();
    let sum = 0;
    for (let digit of strNum) {
        sum += Math.pow(parseInt(digit), 3);
    }
    return sum === num;
}

function findArmstrongNumbers() {
    let armstrongNumbers = [];
    for (let num = 100; num <= 999; num++) {
        if (isArmstrong(num)) {
            armstrongNumbers.push(num);
        }
    }
    return armstrongNumbers;
}

// Example usage:
console.log(findArmstrongNumbers());
// Expected output: [153, 370, 371, 407]
```

10: Construct the following pattern using nested for loops

```
function constructPattern() {
    let pattern = '';
    for (let i = 1; i <= 5; i++) {
        for (let j = 1; j <= i; j++) {
            pattern += '* ';
        }
        pattern += '\n';
    }
    console.log(pattern);
}

// Example usage:
constructPattern();
// Expected output:
// *
// * *
// * * *
// * * * *
// * * * * *
```

11: Compute the greatest common divisor (GCD) of two positive integers

```
function gcd(num1, num2) {
    while (num2 !== 0) {
        let temp = num2;
        num2 = num1 % num2;
        num1 = temp;
    }
    return num1;
}

// Example usage:
console.log(gcd(24, 36)); // Expected output: 12
```

12: Sum the multiples of 3 and 5 under 1000

```
function sumMultiples() {
    let sum = 0;
    for (let i = 1; i < 1000; i++) {
        if (i % 3 === 0 || i % 5 === 0) {
            sum += i;
        }
    }
    console.log(`Sum of multiples of 3 and 5 under 1000 is: ${sum}`);
}
```

```
// Example usage:
sumMultiples();
// Expected output: Sum of multiples of 3 and 5 under 1000 is: 233168
```

JavaScript array

Problem 1: Check if input is an array

```
function is_array(input) {
    return Array.isArray(input);
}

// Test cases
console.log(is_array('w3resource'));  // false
console.log(is_array([1, 2, 4, 0]));   // true
```

Problem 2: Clone an array

```
function array_Clone(arr) {
    return arr.slice(0);
}

// Test cases
console.log(array_Clone([1, 2, 4, 0]));       // [1, 2, 4, 0]
console.log(array_Clone([1, 2, [4, 0]]));    // [1, 2, [4, 0]]
```

Problem 3: Get the first 'n' elements of an array

```
function first(arr, n = 1) {
    if (n < 0) {
        return [];
    }
    return arr.slice(0, n);
}

// Test cases
console.log(first([7, 9, 0, -2]));        // [7]
console.log(first([], 3));                // []
console.log(first([7, 9, 0, -2], 3));     // [7, 9, 0]
console.log(first([7, 9, 0, -2], 6));     // [7, 9, 0, -2]
console.log(first([7, 9, 0, -2], -3));    // []
```

Problem 4: Get the last 'n' elements of an array

```
function last(arr, n = 1) {
    if (n < 0) {
        return [];
    }
    return arr.slice(Math.max(arr.length - n, 0));
}

// Test cases
console.log(last([7, 9, 0, -2]));        // [-2]
console.log(last([7, 9, 0, -2], 3));     // [0, -2]
console.log(last([7, 9, 0, -2], 6));     // [7, 9, 0, -2]
```

Problem 5: Join elements of an array into a string

```
var myColor = ["Red", "Green", "White", "Black"];
console.log(myColor.join());        // "Red,Green,White,Black"
console.log(myColor.join(','));     // "Red,Green,White,Black"
console.log(myColor.join('+'));     // "Red+Green+White+Black"
```

Problem 6: Insert dashes between each two even numbers

```
function insertDashes(num) {
    var str = num.toString();
    var result = [str[0]];

    for (var i = 1; i < str.length; i++) {
        if (str[i-1] % 2 === 0 && str[i] % 2 === 0) {
            result.push('-', str[i]);
        } else {
            result.push(str[i]);
        }
    }
    return result.join('');
}

// Test case
console.log(insertDashes(025468));   // "0-254-6-8"
```

Problem 7: Sort the items of an array

```
var arr1 = [ 3, 8, 7, 6, 5, -4, 3, 2, 1 ];

function sortArray(arr) {
    return arr.sort(function(a, b) {
        return a - b;
    });
}

// Test case
console.log(sortArray(arr1));   // [-4, 1, 2, 3, 3, 5, 6, 7, 8]
```

Problem 8: Find the most frequent item of an array

```
function mostFrequent(arr) {
    if (arr.length === 0) return null;

    var counter = {};
    var maxCount = 0;
    var mostFreq = arr[0];

    for (var i = 0; i < arr.length; i++) {
        var num = arr[i];
        counter[num] = (counter[num] || 0) + 1;
        if (counter[num] > maxCount) {
            maxCount = counter[num];
            mostFreq = num;
        }
    }

    return mostFreq + " ( " + maxCount + " times )";
}

// Test case
var arr1 = [3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];
console.log(mostFrequent(arr1));   // "a ( 5 times )"
```

Problem 9: Swap the case of each character in a string

```
function swapCase(str) {
    var swapped = [];

    for (var i = 0; i < str.length; i++) {
        var char = str[i];
        if (char === char.toUpperCase()) {
            swapped.push(char.toLowerCase());
        } else {
            swapped.push(char.toUpperCase());
        }
    }

    return swapped.join('');
}

// Test case
console.log(swapCase('The Quick Brown Fox'));   // "tHE qUICK bROWN fOX"
```

Problem 10: Print elements of a nested array using nested for loops

```
var a = [[1, 2, 1, 24], [8, 11, 9, 4], [7, 0, 7, 27], [7, 4, 28, 14], [3, 10,
26, 7]];

for (var i = 0; i < a.length; i++) {
    console.log("row " + i);
    for (var j = 0; j < a[i].length; j++) {
        console.log(" " + a[i][j]);
    }
}
```

Problem 11: Find the sum of squares of a numeric vector

```
function sumOfSquares(arr) {
    var sum = 0;
    for (var i = 0; i < arr.length; i++) {
        sum += arr[i] * arr[i];
    }
    return sum;
}

// Test case
console.log(sumOfSquares([1, 2, 3, 4]));   // 30 (1^2 + 2^2 + 3^2 + 4^2 = 30)
```

Problem 12: Compute the sum and product of an array of integers

```
function sumAndProduct(arr) {
    var sum = 0;
    var product = 1;

    for (var i = 0; i < arr.length; i++) {
        sum += arr[i];
        product *= arr[i];
    }

    return {
        sum: sum,
        product: product
    };
}

// Test case
console.log(sumAndProduct([1, 2, 3, 4]));   // { sum: 10, product: 24 }
```

Problem 13: Add items to an empty array and display them

```
function addItems() {
    var arr = [];
    arr.push.apply(arr, arguments); // Adds all arguments to the array
    console.log(arr);
}

// Test case
addItems(1, 2, 3, 'a', 'b');   // [1, 2, 3, "a", "b"]
```

Problem 14: Remove duplicate items from an array (ignore case sensitivity)

```
function removeDuplicates(arr) {
    var uniqueArray = [];
    var seen = {};

    for (var i = 0; i < arr.length; i++) {
        var item = arr[i].toLowerCase();
        if (!seen[item]) {
            seen[item] = true;
            uniqueArray.push(arr[i]);
        }
    }

    return uniqueArray;
}

// Test case
var arr = [NaN, 0, 15, false, -22, '', undefined, 47, null];
console.log(removeDuplicates(arr));   // [NaN, 0, 15, false, -22, '',
undefined, 47, null]
```

Problem 15: Display colors with ordinal positions

```
var color = ["Blue ", "Green", "Red", "Orange", "Violet", "Indigo", "Yellow
"];
var o = ["th","st","nd","rd"];

function displayColors(arr, ordinals) {
    for (var i = 0; i < arr.length; i++) {
        var ordinal = (i % 100 > 10 && i % 100 < 14) ? ordinals[0] :
ordinals[i % 10];
        console.log((i + 1) + ordinal + " choice is " + arr[i] + ".");
    }
}

// Test case
displayColors(color, o);
```

Problem 16: Find leap years in a given range of years

```
function findLeapYears(start, end) {
    var leapYears = [];

    for (var year = start; year <= end; year++) {
        if ((year % 4 === 0 && year % 100 !== 0) || (year % 400 === 0)) {
            leapYears.push(year);
        }
    }

    return leapYears;
}

// Test case
console.log(findLeapYears(2000, 2020));   // [2000, 2004, 2008, 2012, 2016,
2020]
```

Problem 17: Shuffle an array

```
function shuffleArray(arr) {
    var shuffled = arr.slice(0);
    for (var i = shuffled.length - 1; i > 0; i--) {
        var j = Math.floor(Math.random() * (i + 1));
        var temp = shuffled[i];
        shuffled[i] = shuffled[j];
        shuffled[j] = temp;
    }
    return shuffled;
}

// Test case
var arr = [1, 2, 3, 4, 5, 6, 7, 8, 9];
console.log(shuffleArray(arr));
```

Problem 18: Perform a binary search

```
function binarySearch(arr, key) {
    var low = 0;
    var high = arr.length - 1;

    while (low <= high) {
        var mid = Math.floor((low + high) / 2);
        if (arr[mid] === key) {
            return mid;
        } else if (arr[mid] < key) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
    return -1; // Key not found
}

// Test case
var items = [1, 2, 3, 4, 5, 7, 8, 9];
console.log(binarySearch(items, 1));   // 0
console.log(binarySearch(items, 5));   // 4
console.log(binarySearch(items, 6));   // -1 (not found)
```

Problem 19: Compute the sum of each individual index value from two arrays

```
function sumArrays(arr1, arr2) {
    var result = [];
    var maxLength = Math.max(arr1.length, arr2.length);

    for (var i = 0; i < maxLength; i++) {
        var sum = (arr1[i] || 0) + (arr2[i] || 0);
        result.push(sum);
    }

    return result;
}

// Test case
var array1 = [1, 0, 2, 3, 4];
var array2 = [3, 5, 6, 7, 8, 13];
console.log(sumArrays(array1, array2));   // [4, 5, 8, 10, 12, 13]
```

Problem 20: Find duplicate values in an array

```
function findDuplicates(arr) {
    var duplicates = [];
    var seen = {};

    for (var i = 0; i < arr.length; i++) {
        var item = arr[i];
        if (seen[item]) {
            if (seen[item] === 1) {
                duplicates.push(item);
            }
            seen[item]++;
        } else {
            seen[item] = 1;
        }
    }

    return duplicates;
}

// Test case
```

```
var arr = [1, 2, 3, 4, 2, 5, 6, 1, 7];
console.log(findDuplicates(arr));   // [1, 2]
```

Problem 21: Flatten a nested array

```
function flattenArray(arr, shallow) {
    var flattened = [];

    function flatten(arr) {
        arr.forEach(function(item) {
            if (Array.isArray(item) && !shallow) {
                flatten(item);
            } else {
                flattened.push(item);
            }
        });
    }

    flatten(arr);
    return flattened;
}

// Test cases
console.log(flattenArray([1, [2], [3, [[4]]], [5, 6]]));   // [1, 2, 3, 4, 5,
6]
console.log(flattenArray([1, [2], [3, [[4]]], [5, 6]], true));   // [1, 2, 3,
[[4]], 5, 6]
```

Problem 22: Compute the union of two arrays

```
function unionArrays(arr1, arr2) {
    var union = arr1.slice();

    arr2.forEach(function(item) {
        if (union.indexOf(item) === -1) {
            union.push(item);
        }
    });

    return union;
}

// Test case
console.log(unionArrays([1, 2, 3], [100, 2, 1, 10]));   // [1, 2, 3, 100, 10]
```

Problem 23: Find the difference of two arrays

```
function differenceArrays(arr1, arr2) {
    var difference = [];

    arr1.forEach(function(item) {
        if (arr2.indexOf(item) === -1 && difference.indexOf(item) === -1) {
            difference.push(item);
        }
    });

    arr2.forEach(function(item) {
        if (arr1.indexOf(item) === -1 && difference.indexOf(item) === -1) {
            difference.push(item);
        }
    });

    return difference;
}

// Test cases
console.log(differenceArrays([1, 2, 3], [100, 2, 1, 10]));   // [3, 100, 10]
console.log(differenceArrays([1, 2, 3, 4, 5], [1, [2], [3, [[4]]], [5, 6]]));
// [6]
console.log(differenceArrays([1, 2, 3], [100, 2, 1, 10]));   // [3, 100, 10]
```

Problem 24: Remove 'null', '0', '""', 'false', 'undefined', and 'NaN' values from an array

```
function cleanArray(arr) {
    return arr.filter(function(item) {
        return item !== null && item !== undefined && item !== '' && item !==
false && !isNaN(item);
    });
}

// Test case
var arr = [NaN, 0, 15, false, -22, '', undefined, 47, null];
console.log(cleanArray(arr));   // [15, -22, 47]
```

Problem 25: Sort an array of objects by title value

```
var library = [
    { author: 'Bill Gates', title: 'The Road Ahead', libraryID: 1254 },
    { author: 'Steve Jobs', title: 'Walter Isaacson', libraryID: 4264 },
    { author: 'Suzanne Collins', title: 'Mockingjay: The Final Book of The
Hunger Games', libraryID: 3245 }
];

function sortByTitle(arr) {
    return arr.sort(function(a, b) {
        var titleA = a.title.toLowerCase();
        var titleB = b.title.toLowerCase();
        if (titleA < titleB) return -1;
        if (titleA > titleB) return 1;
        return 0;
    });
}

// Test case
console.log(sortByTitle(library));
```

Problem 26: Find a pair of elements from an array whose sum equals a specific target number

```
function findPairWithSum(arr, target) {
    var pairs = [];

    for (var i = 0; i < arr.length; i++) {
        for (var j = i + 1; j < arr.length; j++) {
            if (arr[i] + arr[j] === target) {
                pairs.push([i, j]);
            }
        }
    }

    return pairs;
}

// Test case
var numbers = [10, 20, 10, 40, 50, 60, 70];
var target = 50;
console.log(findPairWithSum(numbers, target));   // Output: [[0, 3], [1, 2]]
```

Problem 27: Retrieve the value of a given property from all elements in an array

```
function getPropertyValues(arr, prop) {
    var values = [];

    arr.forEach(function(obj) {
        if (obj.hasOwnProperty(prop)) {
            values.push(obj[prop]);
        }
    });

    return values;
}

// Test case
var arr = [
    { id: 1, name: 'John', age: 30 },
    { id: 2, name: 'Jane', age: 25 },
    { id: 3, name: 'Doe', age: 40 }
];
```

```
console.log(getPropertyValues(arr, 'name'));   // Output: ['John', 'Jane',
'Doe']
```

Problem 28: Find the longest common starting substring in a set of strings

```
function longestCommonStartingSubstring(arr) {
    if (!arr.length) return '';

    var firstStr = arr[0];
    var maxPrefix = '';

    for (var i = 1; i <= firstStr.length; i++) {
        var prefix = firstStr.slice(0, i);
        if (arr.every(function(str) { return str.startsWith(prefix); })) {
            maxPrefix = prefix;
        } else {
            break;
        }
    }

    return maxPrefix;
}

// Test case
console.log(longestCommonStartingSubstring(['go', 'google']));   // Output:
'go'
```

Problem 29: Fill an array with values (numeric or string with one character) on supplied bounds

```
function numStringRange(start, end, step) {
    var arr = [];
    for (var i = start; i <= end; i += step) {
        arr.push(i);
    }
    return arr;
}

// Test case
console.log(numStringRange('a', 'z', 2));   // Output: ['a', 'c', 'e', 'g',
'i', 'k', 'm', 'o', 'q', 's', 'u', 'w', 'y']
```

Problem 30: Merge two arrays and remove duplicate elements

```
function mergeArrays(arr1, arr2) {
    var merged = arr1.concat(arr2);
    var uniqueArray = [];
    merged.forEach(function(item) {
        if (uniqueArray.indexOf(item) === -1) {
            uniqueArray.push(item);
        }
    });

    return uniqueArray;
}

// Test case
var array1 = [1, 2, 3];
var array2 = [2, 30, 1];
console.log(mergeArrays(array1, array2));   // Output: [1, 2, 3, 30]
```

Problem 31: Remove a specific element from an array

```
function removeArrayElement(arr, element) {
    return arr.filter(function(item) {
        return item !== element;
    });
}

// Test case
console.log(removeArrayElement([2, 5, 9, 6], 5));   // Output: [2, 9, 6]
```

Problem 32: Check if an array contains a specific element

```
function arrayContainsElement(arr, element) {
    return arr.includes(element);
}

// Test case
console.log(arrayContainsElement([2, 5, 9, 6], 5));   // Output: true
console.log(arrayContainsElement([2, 5, 9, 6], 3));   // Output: false
```

Problem 33: Empty an array while keeping the original reference

```
function emptyArray(arr) {
    arr.length = 0;
    return arr;
}

// Test case
var arr = [1, 2, 3, 4];
console.log(emptyArray(arr));   // Output: []
```

Problem 34: Find the nth largest element from an unsorted array

```
function nthLargest(arr, n) {
    if (n > arr.length || n <= 0) {
        return 'Invalid input';
    }

    arr.sort(function(a, b) {
        return b - a;
    });

    return arr[n - 1];
}

// Test case
console.log(nthLargest([43, 56, 23, 89, 88, 90, 99, 652], 4));   // Output:
89
```

Problem 35: Get a random item from an array

```
function getRandomItem(arr) {
    var randomIndex = Math.floor(Math.random() * arr.length);
    return arr[randomIndex];
}

// Test case
var arr = [1, 2, 3, 4, 5];
console.log(getRandomItem(arr));   // Output: (random number from the array)
```

Problem 36: Create an array filled with specified number of elements with pre-filled numeric values

```
function arrayFilled(length, value) {
    return Array(length).fill(value);
}

// Test cases
console.log(arrayFilled(6, 0));   // Output: [0, 0, 0, 0, 0, 0]
console.log(arrayFilled(4, 11));  // Output: [11, 11, 11, 11]
```

Problem 37: Create an array filled with specified number of elements with pre-filled string values

```
function stringArrayFilled(length, value) {
    return Array(length).fill(value);
}
```

```
// Test cases
console.log(stringArrayFilled(3, 'default value'));   // Output: ['default
value', 'default value', 'default value']
console.log(stringArrayFilled(4, 'password'));        // Output: ['password',
'password', 'password', 'password']
```

Problem 38: Move an array element from one position to another

```
function moveArrayElement(arr, fromIndex, toIndex) {
    var element = arr.splice(fromIndex, 1)[0];
    arr.splice(toIndex, 0, element);
    return arr;
}

// Test cases
console.log(moveArrayElement([10, 20, 30, 40, 50], 0, 2));   // Output: [20,
30, 10, 40, 50]
console.log(moveArrayElement([10, 20, 30, 40, 50], -1, -2)); // Output: [10,
20, 30, 50, 40]
```

Problem 39: Filter false, null, 0, blank values from an array

```
function filterArrayValues(arr) {
    return arr.filter(function(item) {
        return item !== false && item !== null && item !== 0 && item !== '';
    });
}

// Test case
var arr = [58, '', 'abcd', true, null, false, 0];
console.log(filterArrayValues(arr));   // Output: [58, 'abcd', true]
```

Problem 40: Generate an array of specified length, filled with integer numbers increasing by one from starting position

```
function arrayRange(start, length) {
    return Array.from({ length: length }, function(_, index) {
        return start + index;
    });
}

// Test cases
console.log(arrayRange(1, 4));   // Output: [1, 2, 3, 4]
console.log(arrayRange(-6, 4));  // Output: [-6, -5, -4, -3]
```

Problem 41: Generate an array between two integers with 1 step length

```
function rangeBetween(start, end) {
    var arr = [];
    for (var i = start; i <= end; i++) {
        arr.push(i);
    }
    return arr;
}

// Test cases
console.log(rangeBetween(4, 7));   // Output: [4, 5, 6, 7]
console.log(rangeBetween(-4, 7));  // Output: [-4, -3, -2, -1, 0, 1, 2, 3, 4,
5, 6, 7]
```

Problem 42: Find unique elements from two arrays

```
function uniqueElements(arr1, arr2) {
    var combinedArray = arr1.concat(arr2);
    var uniqueArray = [...new Set(combinedArray)];
    return uniqueArray;
}

// Test cases
console.log(uniqueElements([1, 2, 3], [100, 2, 1, 10]));   // Output: [1, 2,
3, 100, 10]
console.log(uniqueElements([1, 2, 3, 4, 5], [1, [2], [3, [[4]]], [5, 6]]));
// Output: [1, 2, 3, 4, 5, 6]
console.log(uniqueElements([1, 2, 3], [100, 2, 1, 10]));   // Output: [1, 2,
3, 100, 10]
```

JavaScript Date

1: Check if input is a date object

```
function is_date(input) {
    return input instanceof Date && !isNaN(input);
}

console.log(is_date("October 13, 2014 11:13:00"));  // false
console.log(is_date(new Date(86400000)));            // true
console.log(is_date(new Date(99,5,24,11,33,30,0)));  // true
console.log(is_date([1, 2, 4, 0]));                  // false
```

2: Get the current date with a specified separator

```
function curday(separator) {
    let today = new Date();
    let dd = today.getDate();
    let mm = today.getMonth() + 1;
    let yyyy = today.getFullYear();

    if (dd < 10) {
        dd = '0' + dd;
    }

    if (mm < 10) {
        mm = '0' + mm;
    }

    return mm + separator + dd + separator + yyyy;
}

console.log(curday('/'));  // "11/13/2014"
console.log(curday('-'));  // "11-13-2014"
```

3: Get the number of days in a month

```
function getDaysInMonth(month, year) {
    return new Date(year, month, 0).getDate();
}

console.log(getDaysInMonth(1, 2012));  // 31
console.log(getDaysInMonth(2, 2012));  // 29 (leap year)
console.log(getDaysInMonth(9, 2012));  // 30
console.log(getDaysInMonth(12, 2012)); // 31
```

4: Get the month name from a date

```
function month_name(dt) {
    const months = ["January", "February", "March", "April", "May", "June",
                    "July", "August", "September", "October", "November",
"December"];
    return months[dt.getMonth()];
}

console.log(month_name(new Date("10/11/2009")));   // "October"
console.log(month_name(new Date("11/13/2014")));  // "November"
```

5: Compare dates (greater than, less than, or equal)

```
function compare_dates(date1, date2) {
    if (date1 > date2) return "Date1 > Date2";
    else if (date1 < date2) return "Date1 < Date2";
    else return "Date1 = Date2";
}

console.log(compare_dates(new Date('11/14/2013 00:00'), new Date('11/14/2013 
00:00')));  // "Date1 = Date2"
console.log(compare_dates(new Date('11/14/2013 00:01'), new Date('11/14/2013 
00:00')));  // "Date1 > Date2"
console.log(compare_dates(new Date('11/14/2013 00:00'), new Date('11/14/2013 
00:01')));  // "Date1 < Date2"
```

6: Add specified minutes to a Date object

```
function add_minutes(dt, minutes) {
    return new Date(dt.getTime() + minutes * 60000);
}

console.log(add_minutes(new Date(2014, 10, 2), 30).toString());
// "Sun Nov 02 2014 00:30:00 GMT+0530 (India Standard Time)"
```

7: Test whether a date is a weekend

```
function is_weekend(date) {
    const dt = new Date(date);
    if (dt.getDay() === 0 || dt.getDay() === 6) {
        return "weekend";
    }
    // For clarity, return undefined explicitly if it's not a weekend
}

console.log(is_weekend('Nov 15, 2014'));  // "weekend"
console.log(is_weekend('Nov 16, 2014'));  // "weekend"
console.log(is_weekend('Nov 17, 2014'));  // undefined
```

8: Get difference between two dates in days

```
function date_diff_indays(date1, date2) {
    const diffTime = Math.abs(date2 - date1);
    const diffDays = Math.ceil(diffTime / (1000 * 60 * 60 * 24));
    return diffDays;
}

console.log(date_diff_indays(new Date('04/02/2014'), new 
Date('11/04/2014')));  // 216
console.log(date_diff_indays(new Date('12/02/2014'), new 
Date('11/04/2014')));  // 28
```

9: Get the last day of a month

```
function lastday(year, month) {
    return new Date(year, month + 1, 0).getDate();
}

console.log(lastday(2014, 0));   // 31 (January)
console.log(lastday(2014, 1));   // 28 (February, non-leap year)
console.log(lastday(2014, 11));  // 31 (December)
```

10: Calculate 'yesterday day'

```
function yesterday(date) {
    const dt = new Date(date);
    dt.setDate(dt.getDate() - 1);
    return dt.toString();
}

console.log(yesterday('Nov 15, 2014'));
console.log(yesterday('Nov 16, 2015'));
console.log(yesterday('Nov 17, 2016'));
```

11: Get the maximum date from an array of dates

```
function max_date(dates) {
    return new Date(Math.max.apply(null, dates.map(function(e) {
        return new Date(e);
    }))).toISOString().split('T')[0];
}

console.log(max_date(['2015/02/01', '2015/02/02', '2015/01/03']));  //
"2015/02/02"
```

12: Get the minimum date from an array of dates

```
function min_date(dates) {
    return new Date(Math.min.apply(null, dates.map(function(e) {
        return new Date(e);
    }))).toISOString().split('T')[0];
}

console.log(min_date(['2015/02/01', '2015/02/02', '2015/01/03']));  //
"2015/01/03"
```

13: Return the number of minutes in hours and minutes

```
function timeConvert(minutes) {
    const hours = Math.floor(minutes / 60);
    const mins = minutes % 60;
    return `${minutes} minutes = ${hours} hour(s) and ${mins} minute(s).`;
}

console.log(timeConvert(200));  // "200 minutes = 3 hour(s) and 20
minute(s)."
```

14: Get the amount of days in a year

```
function days_of_a_year(year) {
    return isLeapYear(year) ? 366 : 365;
}

function isLeapYear(year) {
    return (year % 4 === 0 && year % 100 !== 0) || year % 400 === 0;
}

console.log(days_of_a_year(2015));  // 365
console.log(days_of_a_year(2016));  // 366
```

15: Get the quarter (1 to 4) of the year

```
function quarter_of_the_year(date) {
    const month = date.getMonth() + 1;
    return Math.ceil(month / 3);
}

console.log(quarter_of_the_year(new Date(2015, 1, 21)));   // 2
console.log(quarter_of_the_year(new Date(2015, 10, 18)));  // 4
```

16: Count the number of days passed since the beginning of the year

```
function days_passed(date) {
    const startOfYear = new Date(date.getFullYear(), 0, 0);
    const diff = date - startOfYear;
    const oneDay = 1000 * 60 * 60 * 24;
    return Math.floor(diff / oneDay);
}

console.log(days_passed(new Date(2015, 0, 15)));   // 15
console.log(days_passed(new Date(2015, 11, 14)));  // 348
```

17: Convert a Unix timestamp to time

```
function unix_timestamp_to_time(timestamp) {
    const dt = new Date(timestamp * 1000);
    const hours = dt.getHours().toString().padStart(2, '0');
    const minutes = dt.getMinutes().toString().padStart(2, '0');
    const seconds = dt.getSeconds().toString().padStart(2, '0');
    return `${hours}:${minutes}:${seconds}`;
}

console.log(unix_timestamp_to_time(1615481504));  // "12:18:24"
```

18: Calculate age from a birth date

```
function calculate_age(birthDate) {
    const today = new Date();
    const diff = today - birthDate;
    const ageDate = new Date(diff);
    return Math.abs(ageDate.getUTCFullYear() - 1970);
}

console.log(calculate_age(new Date(1982, 11, 4)));   // 41
console.log(calculate_age(new Date(1962, 1, 1)));    // 62
```

19: Get the day of the month, 2 digits with leading zeros

```
function day_of_the_month(date) {
    return ('0' + date.getDate()).slice(-2);
}

const d = new Date(2015, 10, 1);
console.log(day_of_the_month(d));  // "01"
```

20: Get a textual representation of a day (three letters, Mon through Sun)

```
function short_Days(date) {
    const days = ['Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat'];
    return days[date.getDay()];
}

const dt = new Date(2015, 10, 1);
console.log(short_Days(dt));  // "Sun"
```

21: Get a full textual representation of the day of the week (Sunday through Saturday)

```
function long_Days(date) {
    const days = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday',
'Friday', 'Saturday'];
    return days[date.getDay()];
}

const dt = new Date(2015, 10, 1);
console.log(long_Days(dt));  // "Sunday"
```

22: Get ISO-8601 numeric representation of the day of the week (1 for Monday to 7 for Sunday)

```
function ISO_numeric_date(date) {
    return date.getDay() === 0 ? 7 : date.getDay();
}

const dt = new Date(2015, 10, 1);
console.log(ISO_numeric_date(dt));  // 7
```

23: Get English ordinal suffix for the day of the month (1st, 2nd, 3rd, or 4th)

```
function english_ordinal_suffix(date) {
    const day = date.getDate();
    if (day % 10 === 1 && day !== 11) {
        return day + "st";
    } else if (day % 10 === 2 && day !== 12) {
        return day + "nd";
    } else if (day % 10 === 3 && day !== 13) {
        return day + "rd";
    } else {
        return day + "th";
    }
}

const dt = new Date(2015, 10, 1);
console.log(english_ordinal_suffix(dt));  // "1st"
```

24: Get ISO-8601 week number of year (weeks starting on Monday)

```
function ISO8601_week_no(date) {
    const dt = new Date(date);
    dt.setHours(0, 0, 0);
    dt.setDate(dt.getDate() + 4 - (dt.getDay() || 7));
    const yearStart = new Date(dt.getFullYear(), 0, 1);
    const weekNumber = Math.ceil((((dt - yearStart) / 86400000) + 1) / 7);
    return weekNumber;
}

const dt = new Date(2015, 10, 1);
console.log(ISO8601_week_no(dt));  // 44
```

25: Get a full textual representation of a month (January through December)

```
function full_month(date) {
    const months = ['January', 'February', 'March', 'April', 'May', 'June',
                    'July', 'August', 'September', 'October', 'November',
'December'];
    return months[date.getMonth()];
}

const dt = new Date(2015, 10, 1);
console.log(full_month(dt));  // "November"
```

26: Get a numeric representation of a month, with leading zeros (01 through 12)

```
function numeric_month(date) {
    return ('0' + (date.getMonth() + 1)).slice(-2);
}

const dt = new Date(2015, 10, 1);
console.log(numeric_month(dt));  // "11"
```

27: Get a short textual representation of a month, three letters (Jan through Dec)

```
function short_months(date) {
    const months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
                    'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'];
    return months[date.getMonth()];
}

const dt = new Date(2015, 10, 1);
console.log(short_months(dt));  // "Nov"
```

28: Get a full numeric representation of a year (4 digits)

```
function full_year(date) {
    return date.getFullYear();
}

const dt = new Date(2015, 10, 1);
```

```
console.log(full_year(dt));  // 2015
```

29: Get a two-digit representation of a year

```
function short_year(date) {
    return ('' + date.getFullYear()).slice(-2);
}

const dt = new Date(1989, 10, 1);
console.log(short_year(dt));  // "89"
```

30: Get lowercase Ante meridiem (AM) and Post meridiem (PM)

```
function lowercase_Meridiem(date) {
    return date.getHours() >= 12 ? 'pm' : 'am';
}

const dt = new Date();
console.log(lowercase_Meridiem(dt));  // "am" or "pm"
```

31: Get uppercase Ante meridiem (AM) and Post meridiem (PM)

```
function uppercase_Meridiem(date) {
    return date.getHours() >= 12 ? 'PM' : 'AM';
}

const dt = new Date();
console.log(uppercase_Meridiem(dt));  // "AM" or "PM"
```

32: Swatch Internet time (000 through 999)

```
function internet_time(date) {
    const secondsInDay = date.getUTCHours() * 3600 + date.getUTCMinutes() *
60 + date.getUTCSeconds();
    const beats = Math.floor((secondsInDay / 86.4) % 1000);
    return beats.toString().padStart(3, '0');
}

const dt = new Date();
console.log(internet_time(dt));  // Example output: "812"
```

33: Get 12-hour format of an hour with leading zeros

```
function hours_with_zeroes(date) {
    return ('0' + (date.getHours() % 12 || 12)).slice(-2);
}

const dt = new Date();
console.log(hours_with_zeroes(dt));  // Example output: "05" (for 5 AM/PM)
```

34: Get 24-hour format of an hour without leading zeros

```
function hours_without_zeroes(date) {
    return date.getHours();
}

const dt = new Date();
console.log(hours_without_zeroes(dt));  // Example output: 5 (for 5 AM/PM)
```

35: Get minutes with leading zeros (00 to 59)

```
function minutes_with_leading_zeros(date) {
    return ('0' + date.getMinutes()).slice(-2);
}

const dt = new Date();
console.log(minutes_with_leading_zeros(dt));  // Example output: "03"
```

36: Get seconds with leading zeros (00 through 59)

```
function seconds_with_leading_zeros(date) {
    return ('0' + date.getSeconds()).slice(-2);
}

const dt = new Date();
console.log(seconds_with_leading_zeros(dt));  // Example output: "09"
```

37: Get Timezone

```
function get_timezone() {
    return Intl.DateTimeFormat().resolvedOptions().timeZone;
}

console.log(get_timezone());  // Example output: "Asia/Kolkata"
```

38: Find whether or not the date is in Daylight Saving Time

```
function daylights_savings(date) {
    const jan = new Date(date.getFullYear(), 0, 1);
    const jul = new Date(date.getFullYear(), 6, 1);
    return date.getTimezoneOffset() < Math.max(jan.getTimezoneOffset(),
jul.getTimezoneOffset()) ? 1 : 0;
}

const dt = new Date();
console.log(daylights_savings(dt));  // Example output: 1 (if in Daylight
Saving Time)
```

39: Get difference to Greenwich Mean Time (GMT) in hours

```
function diff_to_GMT(date) {
    const offset = date.getTimezoneOffset();
    const sign = offset > 0 ? '-' : '+';
    const hours = Math.floor(Math.abs(offset) / 60);
    const minutes = Math.abs(offset) % 60;
    return `${sign}${hours.toString().padStart(2,
'0')}.${minutes.toString().padStart(2, '0')}`;
}

const dt = new Date();
console.log(diff_to_GMT(dt));  // Example output: "+05.30"
```

40: Get timezone offset in seconds

```
function timezone_offset_in_seconds(date) {
    return date.getTimezoneOffset() * 60;
}

const dt = new Date();
console.log(timezone_offset_in_seconds(dt));  // Example output: 19800 (for
UTC+05:30)
```

41: Add specified years to a date

```
function add_years(date, years) {
    const result = new Date(date);
    result.setFullYear(result.getFullYear() + years);
    return result;
}

const dt = new Date(2014, 10, 2);
console.log(add_years(dt, 10).toString());  // Example output: "Sat Nov 02
2024 00:00:00 GMT+0530 (India Standard Time)"
```

42: Add specified weeks to a date

```
function add_weeks(date, weeks) {
    const result = new Date(date);
    result.setDate(result.getDate() + weeks * 7);
    return result;
}

const dt = new Date(2014, 10, 2);
console.log(add_weeks(dt, 10).toString());  // Example output: "Sun Jan 11
2015 00:00:00 GMT+0530 (India Standard Time)"
```

43: Add specified months to a date

```
function add_months(date, months) {
    const result = new Date(date);
    result.setMonth(result.getMonth() + months);
    return result;
}

const dt = new Date(2014, 10, 2);
console.log(add_months(dt, 10).toString());  // Example output: "Wed Sep 02
2015 00:00:00 GMT+0530 (India Standard Time)"
```

44: Get time differences in minutes between two dates

```
function diff_minutes(dt1, dt2) {
    const diff = Math.abs(dt2 - dt1);
    return Math.floor((diff / 1000) / 60);
}

const dt1 = new Date("October 13, 2014 11:11:00");
const dt2 = new Date("October 13, 2014 11:13:00");
console.log(diff_minutes(dt1, dt2));  // Example output: 2
```

45: Get time differences in hours between two dates

```
function diff_hours(dt1, dt2) {
    const diff = Math.abs(dt2 - dt1);
    return Math.floor((diff / (1000 * 60 * 60)));
}

const dt1 = new Date("October 13, 2014 08:11:00");
const dt2 = new Date("October 13, 2014 11:13:00");
console.log(diff_hours(dt1, dt2));  // Example output: 3
```

46: Get time differences in days between two dates

```
function diff_days(dt1, dt2) {
    const diff = Math.abs(dt2 - dt1);
    return Math.floor(diff / (1000 * 60 * 60 * 24));
}

const dt1 = new Date("October 13, 2014 08:11:00");
const dt2 = new Date("October 19, 2014 11:13:00");
console.log(diff_days(dt1, dt2));  // Example output: 6
```

47: Get time differences in weeks between two dates

```
function diff_weeks(dt1, dt2) {
    const diff = Math.abs(dt2 - dt1);
    return Math.floor(diff / (1000 * 60 * 60 * 24 * 7));
}

const dt1 = new Date("June 13, 2014 08:11:00");
const dt2 = new Date("October 19, 2014 11:13:00");
console.log(diff_weeks(dt1, dt2));  // Example output: 18
```

48: Get time differences in months between two dates

```
function diff_months(dt1, dt2) {
    const diff = (dt2.getMonth() - dt1.getMonth()) +
                 (12 * (dt2.getFullYear() - dt1.getFullYear()));
    return diff;
}

const dt1 = new Date("June 13, 2014 08:11:00");
const dt2 = new Date("October 19, 2014 11:13:00");
console.log(diff_months(dt1, dt2));  // Example output: 5
```

49: Get time differences in years between two dates

```
function diff_years(dt1, dt2) {
    const diff = dt2.getFullYear() - dt1.getFullYear();
    return diff;
}

const dt1 = new Date("June 13, 2014 08:11:00");
const dt2 = new Date("October 19, 2017 11:13:00");
console.log(diff_years(dt1, dt2));  // Example output: 3
```

50: Get the week start date

```
function get_week_start(date) {
    const day = date.getDay();
    const diff = date.getDate() - day + (day === 0 ? -6 : 1); // adjust when
day is Sunday
    return new Date(date.setDate(diff));
}

const dt = new Date(); // current date
console.log(get_week_start(dt)); // Example output: Start date of the current
week
```

51: Get the week end date

```
function get_week_end(date) {
    const day = date.getDay();
    const diff = date.getDate() + 6 - day; // end of the week is 6 days after
the start
    return new Date(date.setDate(diff));
}

const dt = new Date(); // current date
console.log(get_week_end(dt)); // Example output: End date of the current
week
```

52: Get the month start date

```
function get_month_start(date) {
    return new Date(date.getFullYear(), date.getMonth(), 1);
}

const dt = new Date(); // current date
console.log(get_month_start(dt)); // Example output: Start date of the
current month
```

53: Get the month end date

```
function get_month_end(date) {
    return new Date(date.getFullYear(), date.getMonth() + 1, 0);
}

const dt = new Date(); // current date
console.log(get_month_end(dt)); // Example output: End date of the current
month
```

JavaScript String

1. **Check if input is a string:**

```
function is_string(input) {
    return typeof input === 'string';
}
```

Test:

```
console.log(is_string('w3resource')); // true
console.log(is_string([1, 2, 4, 0])); // false
```

2. **Check if a string is blank:**

```
function is_Blank(input) {
    return input.trim().length === 0;
}
```

Test:

```
console.log(is_Blank('')); // true
console.log(is_Blank('abc')); // false
```

3. **Split a string into an array of words:**

```
function string_to_array(input) {
    return input.trim().split(/\s+/);
}
```

Test:

```
console.log(string_to_array("Robin Singh")); // ["Robin", "Singh"]
```

4. **Remove specified number of characters from a string:**

```
function truncate_string(input, n) {
    return input.slice(0, n);
}
```

Test:

```
console.log(truncate_string("Robin Singh", 4)); // "Robi"
```

5. **Convert a string into abbreviated form:**

```
function abbrev_name(input) {
    let names = input.trim().split(' ');
    if (names.length > 1) {
        return `${names[0]} ${names[names.length - 1][0]}.`;
    }
    return input;
}
```

Test:

```
console.log(abbrev_name("Robin Singh")); // "Robin S."
```

6. **Hide email addresses to protect from unauthorized users:**

```
function protect_email(email) {
    let [name, domain] = email.split('@');
    let masked_name = name.slice(0, Math.floor(name.length / 2)) +
'...';
    return masked_name + '@' + domain;
}
```

Test:

```
console.log(protect_email("robin_singh@example.com")); //
"robin...@example.com"
```

7. **Parameterize a string:**

```
function string_parameterize(input) {
    return input.trim().toLowerCase().replace(/[^a-zA-Z0-9 -]/g,
'').replace(/\s+/g, '-');
}
```

Test:

```
console.log(string_parameterize("Robin Singh from USA.")); // "robin-
singh-from-usa"
```

8. **Capitalize the first letter of a string:**

```
function capitalize(input) {
    return input.charAt(0).toUpperCase() + input.slice(1);
}
```

Test:

```
console.log(capitalize('js string exercises')); // "Js string
exercises"
```

9. **Capitalize the first letter of each word in a string:**

```
function capitalize_Words(input) {
    return input.toLowerCase().split(' ').map(word =>
word.charAt(0).toUpperCase() + word.slice(1)).join(' ');
}
```

Test:

```
console.log(capitalize_Words('js string exercises')); // "Js String
Exercises"
```

10. **Swap case of a string:**

```
function swapcase(input) {
    return input.split('').map(char => {
        if (char === char.toUpperCase()) {
            return char.toLowerCase();
        } else {
            return char.toUpperCase();
        }
    }).join('');
}
```

Test:

```
console.log(swapcase('AaBbc')); // "aAbBC"
```

11. **Convert a string into camel case:**

```
function camelize(input) {
    return input.replace(/(?:^\w|[A-Z]|\b\w)/g, (word, index) => index
=== 0 ? word.toLowerCase() : word.toUpperCase()).replace(/\s+/g, '');
}
```

Test:

```
console.log(camelize("  Exercises")); // " Exercises"
```

12. **Uncamelize a string:**

```
function uncamelize(input, separator) {
    separator = typeof separator === 'undefined' ? ' ' : separator;
    return input.replace(/([a-z])([A-Z])/g, '$1' + separator +
'$2').toLowerCase();
}
```

Test:

```
console.log(uncamelize('helloWorld')); // "hello world"
console.log(uncamelize('helloWorld', '-')); // "hello-world"
console.log(uncamelize('helloWorld', '_')); // "hello_world"
```

13. **Concatenate a string n times:**

```
function repeat(input, n = 1) {
    return input.repeat(n);
}
```

Test:

```
console.log(repeat('Ha!')); // "Ha!"
console.log(repeat('Ha!', 2)); // "Ha!Ha!"
console.log(repeat('Ha!', 3)); // "Ha!Ha!Ha!"
```

14. **Insert a string within a string at a particular position:**

```
function insert(main_string, ins_string, pos = 0) {
    if (typeof pos === 'undefined') {
        pos = 0;
    }
    if (typeof ins_string === 'undefined') {
        ins_string = '';
    }
    return main_string.slice(0, pos) + ins_string +
main_string.slice(pos);
}
```

Test:

```
console.log(insert('We are doing some exercises.')); // "We are doing
some exercises."
console.log(insert('We are doing some exercises.', '  ')); // "  We are
doing some exercises."
console.log(insert('We are doing some exercises.', '  ', 18)); // "We
are doing some   exercises."
```

15. **Humanize number with correct suffix:**

```
function humanize_format(num) {
    if (typeof num === 'undefined') return;
    let suffix = '';
    if (num % 100 >= 11 && num % 100 <= 13) {
        suffix = 'th';
    } else {
        switch (num % 10) {
            case 1:
                suffix = 'st';
                break;
            case 2:
                suffix = 'nd';
                break;
            case 3:
                suffix = 'rd';
                break;
            default:
                suffix = 'th';
                break;
        }
    }
    return num + suffix;
}
```

Test:

```
console.log(humanize_format()); // undefined
console.log(humanize_format(1)); // "1st"
console.log(humanize_format(8)); // "8th"
console.log(humanize_format(301)); // "301st"
console.log(humanize_format(402)); // "402nd"
```

16. **Truncate a string if longer than specified number of characters:**

```
function text_truncate(input, length = input.length, ending = '...') {
    if (length >= input.length) return input;
    return input.slice(0, length) + ending;
}
```

Test:

```
console.log(text_truncate('We are doing JS string exercises.')); // "We
are doing JS string exercises."
console.log(text_truncate('We are doing JS string exercises.', 19)); //
"We are doing JS ..."
console.log(text_truncate('We are doing JS string exercises.', 15,
'!!')); // "We are doing !!"
```

17. **Chop a string into chunks of given length:**

```
function string_chop(input, chunk_size = 1) {
    let result = [];
    for (let i = 0; i < input.length; i += chunk_size) {
        result.push(input.slice(i, i + chunk_size));
    }
    return result;
}
```

Test:

```
console.log(string_chop('w3resource')); // ["w3resource"]
console.log(string_chop('w3resource', 2)); // ["w3", "re", "so", "ur",
"ce"]
console.log(string_chop('w3resource', 3)); // ["w3r", "eso", "urc",
"e"]
```

18. **Count the occurrence of a substring in a string:**

```
function count(main_str, sub_str, case_sensitive = true) {
    if (!case_sensitive) {
        main_str = main_str.toLowerCase();
        sub_str = sub_str.toLowerCase();
    }
    return main_str.split(sub_str).length - 1;
}
```

Test:

```
console.log(count("The quick brown fox jumps over the lazy dog",
'the')); // 2
console.log(count("The quick brown fox jumps over the lazy dog", 'fox',
false)); // 1
```

19. **Escape an HTML string:**

```
function escape_HTML(html_str) {
    return html_str.replace(/[&<>"']/g, tag => ({
        '&': '&amp;',
        '<': '&
```

20. **Pad (left, right) a string to get to a determined length:**

```
function formatted_string(pad, input, type = 'l') {
    if (typeof input === 'undefined') return;
    let padLength = pad.length;
    let inputStr = input.toString();
    switch (type) {
        case 'l':
            return (pad + inputStr).slice(-padLength);
        case 'r':
            return (inputStr + pad).substring(0, padLength);
        default:
            return inputStr;
    }
}
```

Test:

```
console.log(formatted_string('0000', 123, 'l')); // "0123"
console.log(formatted_string('00000000', 123, 'r')); // "12300000"
```

21. **Repeat a string a specified number of times:**

```
function repeat_string(input, count) {
    if (typeof input !== 'string' || count <= 0) {
        return "Error in string or count.";
    }
    return input.repeat(count);
}
```

Test:

```
console.log(repeat_string('a', 4)); // "aaaa"
console.log(repeat_string('a')); // "Error in string or count."
```

22. **Get a part of a string after a specified character:**

```
function subStrAfterChars(input, char, pos) {
    if (!pos) return input.split(char)[0];
    else return input.split(char)[1];
}
```

Test:

```
console.log(subStrAfterChars('w3resource:   Exercises', ':', 'a')); //
"w3resource"
console.log(subStrAfterChars('w3resource:   Exercises', 'E', 'b')); //
"xercises"
```

23. **Strip leading and trailing spaces from a string:**

```
function strip(input) {
    return input.trim();
}
```

Test:

```
console.log(strip('w3resource ')); // "w3resource"
console.log(strip(' w3resource')); // "w3resource"
console.log(strip(' w3resource ')); // "w3resource"
```

24. **Truncate a string to a certain number of words:**

```
function truncate(input, n) {
    return input.split(' ').splice(0, n).join(' ');
}
```

Test:

```
console.log(truncate('The quick brown fox jumps over the lazy dog',
4)); // "The quick brown fox"
```

25. **Alphabetize a given string:**

```
function alphabetize_string(input) {
    return input.split('').sort((a, b) => a.localeCompare(b)).join('');
}
```

Test:

```
console.log(alphabetize_string('United States')); // "SUadeeinsttt"
```

26. **Remove the first occurrence of a given 'search string' from a string:**

```
function remove_first_occurrence(input, search) {
    let index = input.indexOf(search);
    if (index === -1) return input;
    return input.slice(0, index) + input.slice(index + search.length);
}
```

Test:

```
console.log(remove_first_occurrence("The quick brown fox jumps over the
lazy dog", 'the')); // "The quick brown fox jumps over lazy dog"
```

27. **Convert ASCII to Hexadecimal format:**

```
function ascii_to_hexa(str) {
    let hex = '';
    for (let i = 0; i < str.length; i++) {
        hex += ('0' + str.charCodeAt(i).toString(16)).slice(-2);
    }
    return hex;
}
```

Test:

```
console.log(ascii_to_hexa('12')); // "3132"
console.log(ascii_to_hexa('100')); // "313030"
```

28. **Convert Hexadecimal to ASCII format:**

```
function hex_to_ascii(hex_str) {
    let str = '';
    for (let i = 0; i < hex_str.length; i += 2) {
        str += String.fromCharCode(parseInt(hex_str.substr(i, 2), 16));
    }
    return str;
}
```

Test:

```
console.log(hex_to_ascii('3132')); // "12"
console.log(hex_to_ascii('313030')); // "100"
```

29. **Find a word within a string:**

```
function search_word(input, word) {
    let count = 0;
    input.split(' ').forEach(w => {
        if (w === word) count++;
    });
    return `'${word}' was found ${count} times.`;
}
```

Test:

```
console.log(search_word('The quick brown fox', 'fox')); // "'fox' was
found 1 times."
console.log(search_word('aa, bb, cc, dd, aa', 'aa')); // "'aa' was
found 2 times."
```

30. **Check if a string ends with specified suffix:**

```
function string_endsWith(input, suffix) {
    return input.endsWith(suffix);
}
```

Test:

```
console.log(string_endsWith('JS PHP PYTHON', 'PYTHON')); // true
console.log(string_endsWith('JS PHP PYTHON', '')); // false
```

31. **Escape special characters for HTML:**

```
function escape_html(input) {
    return input.replace(/[&<>"']/g, tag => ({
        '&': '&amp;',
        '<': '&lt;',
        '>': '&gt;',
        '"': '&quot;',
        "'": '&#39;'
    }[tag] || tag));
}
```

Test:

```
console.log(escape_html('PHP & MySQL')); // "PHP &amp; MySQL"
console.log(escape_html('3 > 2')); // "3 &gt; 2"
```

32. **Remove non-printable ASCII characters:**

```
function remove_non_ascii(input) {
    return input.replace(/[^\x20-\x7E]/g, '');
}
```

Test:

```
console.log(remove_non_ascii('äÄçÇéÉêPHP-MySQLöÖÐþúÚ')); // "PHP-MySQL"
```

33. **Remove non-word characters:**

```
function remove_non_word(input) {
    return input.replace(/[^A-Za-z0-9\s]/g, '');
}
```

Test:

```
console.log(remove_non_word('PHP ~!@#$%^&*()+`-={}[]|\\:";\'/?><.,
MySQL')); // "PHP - MySQL"
```

34. **Convert a string to title case:**

```
function sentenceCase(input) {
    return input.toLowerCase().replace(/(^|\s)\S/g, (char) =>
char.toUpperCase());
}
```

Test:

```
console.log(sentenceCase('PHP exercises. python exercises.')); // "Php
Exercises. Python Exercises."
```

35. **Remove HTML/XML tags from a string:**

```
function strip_html_tags(input) {
    return input.replace(/<[^>]*>/g, '');
}
```

Test:

```
console.log(strip_html_tags('<p><strong><em>PHP
Exercises</em></strong></p>')); // "PHP Exercises"
```

36. **Create a Zerofilled value with optional +, - sign:**

```
function zeroFill(num, width, sign) {
    sign = sign || '';
    let numStr = Math.abs(num).toString();
    while (numStr.length < width) numStr = '0' + numStr;
    return (num < 0 ? '-' : sign) + numStr;
}
```

Test:

```
console.log(zeroFill(120, 5, '-')); // "+00120"
console.log(zeroFill(29, 4)); // "0029"
```

37. **Test case insensitive string comparison:**

```
function compare_strings(str1, str2) {
    return str1.toLowerCase() === str2.toLowerCase();
}
```

Test:

```
console.log(compare_strings('abcd', 'AbcD')); // true
console.log(compare_strings('ABCD', 'Abce')); // false
```

38. **Create a case-insensitive search:**

```
function case_insensitive_search(input, search) {
    let regex = new RegExp(search, 'i');
    return regex.test(input) ? 'Matched' : 'Not Matched';
}
```

Test:

```
console.log(case_insensitive_search('  Exercises', 'exercises')); //
"Matched"
console.log(case_insensitive_search('  Exercises', 'Exercises')); //
"Matched"
console.log(case_insensitive_search('  Exercises', 'Exercisess')); //
"Not Matched"
```

JavaScript Validation with regular:

1. Check if first character of a string is uppercase

```
function isFirstCharUppercase(str) {
    return /^[A-Z]/.test(str);
}

// Test
console.log(isFirstCharUppercase("Hello")); // true
console.log(isFirstCharUppercase("world")); // false
```

2. Check a credit card number

```
function isValidCreditCardNumber(cardNumber) {
    // Remove spaces and dashes from the card number
    cardNumber = cardNumber.replace(/\s+/g, '').replace(/-/g, '');

    // Check if the input consists of digits only and is of valid length
    if (!/^\d{13,16}$/.test(cardNumber)) {
        return false;
    }

    // Apply Luhn's algorithm to check validity
    let sum = 0;
    let doubleUp = false;
    for (let i = cardNumber.length - 1; i >= 0; i--) {
        let digit = parseInt(cardNumber.charAt(i), 10);
        if (doubleUp) {
            digit *= 2;
            if (digit > 9) digit -= 9;
        }
        sum += digit;
        doubleUp = !doubleUp;
    }

    return (sum % 10) === 0;
}

// Test
console.log(isValidCreditCardNumber("4111-1111-1111-1111")); // true (Visa)
console.log(isValidCreditCardNumber("5500-0000-0000-0004")); // true
(Mastercard)
console.log(isValidCreditCardNumber("6011000000000004"));   // true
(Discover)
console.log(isValidCreditCardNumber("1234-5678-9012-3456")); // false
(Invalid)
```

3. Pattern to match email addresses

```
const emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;

// Test
console.log(emailPattern.test("test@example.com")); // true
console.log(emailPattern.test("invalid_email@.com")); // false
```

5. Implement a trim function using regular expression

```
function customTrim(str) {
    return str.replace(/^\s+|\s+$/g, '').replace(/\s+/g, ' ');
}

// Test
console.log(customTrim("   Hello    world   ")); // "Hello world"
```

6. Count number of words in a string

```
function countWords(str) {
    // Remove leading/trailing whitespace, replace multiple spaces with
single space
    str = str.trim().replace(/\s+/g, ' ');
    // Split the string by spaces and count the elements
    return str.split(' ').length;
}

// Test
console.log(countWords("Hello    world")); // 2
console.log(countWords("   Hello    world   ")); // 2
```

7. Check whether a given value is an IP address

```
function isIPAddress(value) {
    const ipAddressPattern = /^([0-9]{1,3}\.){3}[0-9]{1,3}$/;
    return ipAddressPattern.test(value);
}

// Test
console.log(isIPAddress("192.168.0.1")); // true
console.log(isIPAddress("256.0.0.1")); // false (invalid)
```

8. Count number of vowels in a string

```
function countVowels(str) {
    const vowelPattern = /[aeiou]/gi;
    const matches = str.match(vowelPattern);
    return matches ? matches.length : 0;
}

// Test
console.log(countVowels("Hello world")); // 3
```

9. Check whether a given value is a valid URL

```
function isValidURL(value) {
    try {
        new URL(value);
        return true;
    } catch (error) {
        return false;
    }
}

// Test
console.log(isValidURL("https://www.example.com")); // true
console.log(isValidURL("not a url")); // false
```

10. Check whether a given value is alphanumeric

```
function isAlphaNumeric(value) {
    return /^[a-zA-Z0-9]+$/.test(value);
}

// Test
console.log(isAlphaNumeric("abc123")); // true
console.log(isAlphaNumeric("abc123!")); // false
```

11. Check whether a given value is a time string

```
function isTimeString(value) {
    // Regular expression to match HH:MM format
    const timePattern = /^([01]?[0-9]|2[0-3]):[0-5][0-9]$/;
    return timePattern.test(value);
}

// Test
console.log(isTimeString("10:30")); // true
console.log(isTimeString("23:59")); // true
console.log(isTimeString("5:00"));  // true
console.log(isTimeString("25:00")); // false (invalid hour)
console.log(isTimeString("10:61")); // false (invalid minute)
console.log(isTimeString("1030"));  // false (missing colon)
```

12. Check whether a given value is a US ZIP code

```
function isUSZipCode(value) {
    const zipCodePattern = /^\d{5}(?:[-\s]\d{4})?$/;
    return zipCodePattern.test(value);
}

// Test
console.log(isUSZipCode("12345"));    // true
console.log(isUSZipCode("12345-6789"));// true
console.log(isUSZipCode("1234"));     // false (too short)
console.log(isUSZipCode("12345-"));   // false (incomplete)
console.log(isUSZipCode("12345-67890"));// false (too long)
```

13. Check whether a given value is a UK Post Code

```
avascript

function isUKPostCode(value) {
    const ukPostCodePattern = /^(GIR|[A-Z]\d[A-Z\d]?|[A-Z]{2}\d{2}|[A-Z]\d[A-
Z]|(?:[A-Z]{2})\d[A-Z]{2})\s?\d[A-Z]{2}$/i;
    return ukPostCodePattern.test(value);
}

// Test
console.log(isUKPostCode("SW1A 1AA"));   // true (London)
console.log(isUKPostCode("M1 1AA"));     // true (Manchester)
console.log(isUKPostCode("W1A 1AA"));    // true (Westminster)
console.log(isUKPostCode("AB1 2CD"));    // true (Aberdeen)
console.log(isUKPostCode("M11AA"));      // true (without space)
console.log(isUKPostCode("12345"));      // false (not a UK postcode format)
```

14. Check whether a given value is a Canada Post Code

```
function isCanadaPostCode(value) {
    const canadaPostCodePattern = /^[ABCEGHJKLMNPRSTVXY]\d[A-Z]\s?\d[A-
Z]\d$/i;
    return canadaPostCodePattern.test(value);
}

// Test
console.log(isCanadaPostCode("K1A 0B1"));   // true (Ottawa)
console.log(isCanadaPostCode("H0H 0H0"));   // true (Santa Claus)
console.log(isCanadaPostCode("12345"));     // false (not a Canadian postal
code format)
```

15. Check whether a given value is a Social Security Number (SSN)

```
function isSSN(value) {
    const ssnPattern = /^\d{3}-\d{2}-\d{4}$/;
    return ssnPattern.test(value);
}

// Test
console.log(isSSN("123-45-6789"));    // true
console.log(isSSN("123456789"));      // false (missing dashes)
console.log(isSSN("123-456-789"));    // false (too many digits)
```

16. Check whether a given value is a hexadecimal value

```
function isHexadecimal(value) {
    const hexPattern = /^[0-9a-fA-F]+$/;
    return hexPattern.test(value);
}

// Test
console.log(isHexadecimal("1a3"));     // true
console.log(isHexadecimal("abc"));     // true
console.log(isHexadecimal("123XYZ"));  // false (contains invalid character
'X')
console.log(isHexadecimal(""));        // false (empty string)
```

17. Check whether a given value is a hex color value

```
function isHexColor(value) {
    const hexColorPattern = /^#?([0-9a-fA-F]{3}|[0-9a-fA-F]{6})$/;
    return hexColorPattern.test(value);
}

// Test
console.log(isHexColor("#f00"));     // true (short form)
console.log(isHexColor("#ff0000"));  // true (long form)
console.log(isHexColor("ff0000"));   // true (without #)
console.log(isHexColor("#abcXYZ"));  // false (contains invalid character
'X')
console.log(isHexColor("#12345"));   // false (not enough digits)
```

18. Check whether a given value represents a domain

```
function isDomain(value) {
    const domainPattern = /^[a-zA-Z0-9-]+(\.[a-zA-Z0-9-]+)*$/;
    return domainPattern.test(value);
}

// Test
console.log(isDomain("example.com"));      // true
console.log(isDomain("subdomain.example.com")); // true
console.log(isDomain("not a domain"));     // false (contains spaces)
console.log(isDomain("123.45.67.89"));     // true (valid IP address, not a
domain)
```

19. Check whether a given value is HTML

```
function isHTML(value) {
    const htmlPattern = /<[a-z][\s\S]*>/i;
    return htmlPattern.test(value);
}

// Test
console.log(isHTML("<p>Hello</p>"));     // true
console.log(isHTML("Plain text"));       // false (no HTML tags)
console.log(isHTML("<div><span>Test</span></div>")); // true
```

20. Check if a given value contains only alpha, dash, and underscore

```
function hasAlphaDashUnderscore(value) {
    const pattern = /^[a-zA-Z-_]+$/;
    return pattern.test(value);
}

// Test
console.log(hasAlphaDashUnderscore("alpha_dash"));   // true
console.log(hasAlphaDashUnderscore("alpha_dash123")); // false (contains
digits)
console.log(hasAlphaDashUnderscore("alpha-dash"));    // true
console.log(hasAlphaDashUnderscore("alpha.dash"));    // false (contains
period)
```

21. Print an integer with commas as thousands separators

```
function thousandsSeparators(num) {
    // Check if num is a number
    if (typeof num !== 'number') {
        return "Invalid input";
    }

    // Convert number to string with commas
    return num.toLocaleString('en-US');
}

// Test
console.log(thousandsSeparators(1000));       // "1,000"
console.log(thousandsSeparators(10000.23));   // "10,000.23"
console.log(thousandsSeparators(100000));     // "100,000"
console.log(thousandsSeparators(1234567.89)); // "1,234,567.89"
```

JavaScript DOM

Task 1: Modify paragraph text style on button click

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>JS DOM paragraph style</title>
<style>
  /* Optional: Default style */
  #text {
    font-family: Arial, sans-serif;
    font-size: 16px;
    color: black;
  }
</style>
<script>
function js_style() {
    // Get the paragraph element
    var text = document.getElementById('text');

    // Modify the style
    text.style.fontFamily = 'Verdana, Geneva, sans-serif';
    text.style.fontSize = '20px';
    text.style.color = 'blue';
}
</script>
</head>
<body>
<p id="text">  Exercises - w3resource</p>
<div>
```

```
  <button id="jsstyle" onclick="js_style()">Style</button>
</div>
</body>
</html>
```

Task 2: Get values of First and Last name from a form

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Return first and last name from a form</title>
<script>
function getFormvalue() {
    var firstName = document.getElementById('fname').value;
    var lastName = document.getElementById('lname').value;
    alert("First Name: " + firstName + "\nLast Name: " + lastName);
}
</script>
</head>
<body>
<form id="form1" onsubmit="getFormvalue(); return false;">
First name: <input type="text" id="fname" value="David"><br>
Last name: <input type="text" id="lname" value="Beckham"><br>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Task 3: Set the background color of a paragraph

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Set background color of a paragraph</title>
<script>
function setBackgroundColor() {
    var paragraph = document.getElementById('text');
    paragraph.style.backgroundColor = 'lightblue';
}
</script>
</head>
<body>
<p id="text">Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
<button onclick="setBackgroundColor()">Set Background Color</button>
</body>
</html>
```

Task 4: Get attributes of a link on button click

```
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>Get attributes of a link</title>
<script>
function getAttributes() {
    var link = document.getElementById('w3r');
    var href = link.getAttribute('href');
    var hreflang = link.getAttribute('hreflang');
    var rel = link.getAttribute('rel');
    var target = link.getAttribute('target');
    var type = link.getAttribute('type');

    alert("href: " + href + "\nhreflang: " + hreflang + "\nrel: " + rel +
"\ntarget: " + target + "\ntype: " + type);
}
</script>
</head>
<body>
<p><a id="w3r" type="text/html" hreflang="en-us" rel="nofollow"
target="_self" href="http://www.w3resource.com/">w3resource</a></p>
<button onclick="getAttributes()">Click here to get attributes value</button>
</body>
</html>
```

Task 5: Add rows to a table

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Insert row in a table</title>
<script>
function insert_Row() {
    var table = document.getElementById('sampleTable');
    var newRow = table.insertRow(table.rows.length);
    var cell1 = newRow.insertCell(0);
    var cell2 = newRow.insertCell(1);
    cell1.innerHTML = 'New row cell 1';
    cell2.innerHTML = 'New row cell 2';
}
</script>
</head>
<body>
<table id="sampleTable" border="1">
<tr><td>Row1 cell1</td><td>Row1 cell2</td></tr>
<tr><td>Row2 cell1</td><td>Row2 cell2</td></tr>
</table>
<input type="button" onclick="insert_Row()" value="Insert row">
</body>
</html>
```

Task 6: Update the content of a specific cell in a table

```
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>Change the content of a cell</title>
<script>
function changeContent() {
    var table = document.getElementById('myTable');
    var row = 1; // Index of row (zero-based)
    var col = 1; // Index of column (zero-based)
    var newText = 'Updated content'; // New content

    table.rows[row].cells[col].innerHTML = newText;
}
</script>
</head>
<body>
<table id="myTable" border="1">
<tr><td>Row1 cell1</td><td>Row1 cell2</td></tr>
<tr><td>Row2 cell1</td><td>Row2 cell2</td></tr>
<tr><td>Row3 cell1</td><td>Row3 cell2</td></tr>
</table>
<input type="button" onclick="changeContent()" value="Change content">
</body>
</html>
```

Task 7: Create a table dynamically based on user input

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Create a table with user input</title>
<script>
function createTable() {
    var rows = parseInt(prompt("Enter number of rows:"));
    var cols = parseInt(prompt("Enter number of columns:"));

    var table = document.getElementById('myTable');
    table.innerHTML = ''; // Clear existing table content

    for (var i = 0; i < rows; i++) {
        var row = table.insertRow(i);
        for (var j = 0; j < cols; j++) {
            var cell = row.insertCell(j);
            cell.innerHTML = "Row-" + i + " Column-" + j;
        }
    }
}
</script>
</head>
<body>
<table id="myTable" border="1">
<!-- Table will be created dynamically -->
</table>
<input type="button" onclick="createTable()" value="Create the table">
</body>
</html>
```

Task 8: Remove items from a dropdown list

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Remove items from a dropdown list</title>
<script>
function removecolor() {
    var select = document.getElementById('colorSelect');
    var index = select.selectedIndex;
    select.remove(index);
}
</script>
</head>
<body>
<form>
<select id="colorSelect">
<option>Red</option>
<option>Green</option>
<option>White</option>
<option>Black</option>
</select>
<input type="button" onclick="removecolor()" value="Select and Remove"><br>
</form>
</body>
</html>
```

Task 9: Count and display items of a dropdown list

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Count and display items of a dropdown list</title>
<script>
function getOptions() {
    var select = document.getElementById('mySelect');
    var count = select.options.length;
    var options = [];
    for (var i = 0; i < count; i++) {
        options.push(select.options[i].text);
    }
    alert("Total items: " + count + "\nItems: " + options.join(', '));
}
</script>
</head>
<body>
<form>
Select your favorite Color :
<select id="mySelect">
<option>Red</option>
<option>Green</option>
<option>Blue</option>
<option>White</option>
```

```
</select>
<input type="button" onclick="getOptions()" value="Count and Output all
items">
</form>
</body>
</html>
```

Task 10: Calculate the volume of a sphere

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Calculate the volume of a sphere</title>
<script>
function calculateVolume() {
    var radius = parseFloat(prompt("Enter the radius of the sphere:"));
    var volume = (4/3) * Math.PI * Math.pow(radius, 3);
    alert("Volume of the sphere with radius " + radius + " units is: " +
volume.toFixed(2) + " cubic units");
}
</script>
</head>
<body>
<input type="button" onclick="calculateVolume()" value="Calculate Sphere
Volume">
</body>
</html>
```

Task 11: Display a random image on button click

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Display a random image</title>
<script>
function displayRandomImage() {
    var images = [
        { src:
"http://farm4.staticflickr.com/3691/11268502654_f28f05966c_m.jpg", width:
"240", height: "160" },
        { src: "http://farm1.staticflickr.com/33/45336904_1aef569b30_n.jpg",
width: "320", height: "195" },
        { src:
"http://farm6.staticflickr.com/5211/5384592886_80a512e2c9.jpg", width: "500",
height: "343" }
    ];

    var randomIndex = Math.floor(Math.random() * images.length);
    var randomImage = images[randomIndex];

    var imgElement = document.createElement('img');
    imgElement.src = randomImage.src;
    imgElement.width = randomImage.width;
    imgElement.height = randomImage.height;

    document.body.appendChild(imgElement);
}
</script>
</head>
<body>
<input type="button" onclick="displayRandomImage()" value="Display Random
Image">
</body>
</html>
```

Task 12: Highlight bold words on mouseover

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Highlight bold words</title>
<script>
function highlightWords() {
    var paragraph = document.getElementById('paragraph');
    var words = paragraph.getElementsByTagName('strong');

    for (var i = 0; i < words.length; i++) {
        words[i].style.backgroundColor = 'yellow';
    }
}
</script>
</head>
<body>
<a href="#" onmouseover="highlightWords()">On mouse over here bold words of
the following paragraph will be highlighted</a>
<p id="paragraph">We have just started this section for the users (beginner
to intermediate) who want to work with various   problems and write scripts
online to test their   skill. <strong>bold words</strong></p>
</body>
</html>
```

Task 13: Get width and height of the window on resize

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Get window width and height on resize</title>
<script>
window.onresize = function() {
    var width = window.innerWidth;
    var height = window.innerHeight;
    alert("Window width: " + width + ", Window height: " + height);
};
</script>
</head>
<body>
```

```
<!-- No UI elements needed for this task -->
</body>
</html>
```

JavaScript Drawing

1. Rectangular Shape

```
const canvas1 = document.getElementById('canvas1');
const ctx1 = canvas1.getContext('2d');

ctx1.fillStyle = 'blue';
ctx1.fillRect(20, 20, 150, 100);
```

2. Circle

```
const canvas2 = document.getElementById('canvas2');
const ctx2 = canvas2.getContext('2d');

ctx2.beginPath();
ctx2.arc(100, 75, 50, 0, 2 * Math.PI);
ctx2.fillStyle = 'red';
ctx2.fill();
```

3. Intersecting Rectangles with Alpha Transparency

```
const canvas3 = document.getElementById('canvas3');
const ctx3 = canvas3.getContext('2d');

ctx3.fillStyle = 'blue';
ctx3.fillRect(20, 20, 100, 100);

ctx3.globalAlpha = 0.5;
ctx3.fillStyle = 'green';
ctx3.fillRect(50, 50, 100, 100);
```

4. Right-Angled Triangle

```
const canvas4 = document.getElementById('canvas4');
const ctx4 = canvas4.getContext('2d');
```

```
ctx4.beginPath();
ctx4.moveTo(50, 50);
ctx4.lineTo(50, 150);
ctx4.lineTo(150, 150);
ctx4.closePath();

ctx4.stroke();
```

5. Diagram using `moveto()` Function

```
const canvas5 = document.getElementById('canvas5');
const ctx5 = canvas5.getContext('2d');

ctx5.beginPath();
ctx5.moveTo(50, 50);
ctx5.lineTo(200, 50);
ctx5.lineTo(125, 150);
ctx5.closePath();

ctx5.stroke();
```

6. Diagonal, White to Black Circles Diagram

```
const canvas6 = document.getElementById('canvas6');
const ctx6 = canvas6.getContext('2d');

const gradient = ctx6.createLinearGradient(0, 0, 200, 200);
gradient.addColorStop(0, 'white');
gradient.addColorStop(1, 'black');

ctx6.beginPath();
ctx6.arc(100, 100, 50, 0, 2 * Math.PI);
ctx6.fillStyle = gradient;
ctx6.fill();
```

JavaScriptObject

1. **List properties of an object:**

```
var student = {
    name: "David Rayy",
    sclass: "VI",
    rollno: 12
};

function listProperties(obj) {
    return Object.keys(obj).join(",");
}

console.log(listProperties(student)); // Output: name,sclass,rollno
```

2. **Delete a property from an object:**

```
var student = {
    name: "David Rayy",
    sclass: "VI",
    rollno: 12
};

console.log("Before deleting:", student);
delete student.rollno;
console.log("After deleting:", student);
```

3. **Get the length of an object (number of properties):**

```
var student = {
    name: "David Rayy",
    sclass: "VI",
    rollno: 12
};

function objectLength(obj) {
    return Object.keys(obj).length;
}

console.log(objectLength(student)); // Output: 3
```

4. **Display reading status of books:**

```
var library = [
    { author: 'Bill Gates', title: 'The Road Ahead', readingStatus: true },
    { author: 'Steve Jobs', title: 'Walter Isaacson', readingStatus: true },
    { author: 'Suzanne Collins', title: 'Mockingjay: The Final Book of The
Hunger Games', readingStatus: false }
];

function displayReadingStatus(library) {
    for (var i = 0; i < library.length; i++) {
        var book = library[i];
        console.log(`${book.title} by ${book.author} - Reading status:
${book.readingStatus}`);
    }
}

displayReadingStatus(library);
```

5. **Calculate the volume of a cylinder using object classes:**

```
class Cylinder {
    constructor(radius, height) {
        this.radius = radius;
        this.height = height;
    }

    getVolume() {
        return (Math.PI * this.radius * this.radius *
this.height).toFixed(4);
    }
}

var cyl = new Cylinder(3, 5);
console.log("Volume of cylinder:", cyl.getVolume()); // Output: Volume of
cylinder: 141.3717
```

6. **Bubble Sort algorithm:**

```
function bubbleSort(arr) {
    var len = arr.length;
    for (var i = 0; i < len - 1; i++) {
        for (var j = 0; j < len - 1 - i; j++) {
            if (arr[j] > arr[j + 1]) {
                var temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    return arr;
}

var data = [6, 4, 0, 3, -2, 1];
console.log("Sorted data:", bubbleSort(data)); // Output: Sorted data: [-2,
0, 1, 3, 4, 6]
```

7. **Return subset of a string:**

```
function substrings(str) {
    var result = [];
    for (var i = 0; i < str.length; i++) {
        for (var j = i + 1; j <= str.length; j++) {
            result.push(str.slice(i, j));
        }
    }
    return result;
}
```

```
console.log(substrings("dog")); // Output: ["d", "do", "dog", "o", "og", "g"]
```

8. **Create a clock that prints every second:**

```
function printTime() {
    var date = new Date();
    var time = date.toLocaleTimeString();
    console.log(time);
}

setInterval(printTime, 1000);
```

9. **Calculate area and perimeter of a circle:**

```
class Circle {
    constructor(radius) {
        this.radius = radius;
    }

    getArea() {
        return (Math.PI * this.radius * this.radius).toFixed(4);
    }

    getPerimeter() {
        return (2 * Math.PI * this.radius).toFixed(4);
    }
}

var circle = new Circle(5);
console.log("Area:", circle.getArea()); // Output: Area: 78.5398
console.log("Perimeter:", circle.getPerimeter()); // Output: Perimeter:
31.4159
```

10. **Sort an array of objects by a property (libraryID):**

```
var library = [
    { title: 'The Road Ahead', author: 'Bill Gates', libraryID: 1254 },
    { title: 'Walter Isaacson', author: 'Steve Jobs', libraryID: 4264 },
    { title: 'Mockingjay: The Final Book of The Hunger Games', author:
'Suzanne Collins', libraryID: 3245 }
];

library.sort((a, b) => a.libraryID - b.libraryID);

console.log(library);
```

11. **Print all methods of an object (Array in this example):**

```
function all_properties(obj) {
    return Object.getOwnPropertyNames(obj).filter(function (property) {
        return typeof obj[property] == 'function';
    });
}

console.log(all_properties(Array)); // Output: ["length", "name",
"prototype", "isArray", "from", "of"]
```

12. **Parse a URL into its components:**

```
function parse_URL(url) {
    var parser = document.createElement('a');
    parser.href = url;

    return {
        protocol: parser.protocol,
        host: parser.host,
        port: parser.port,
        path: parser.pathname,
        query: parser.search,
        hash: parser.hash,
        params: (function () {
            var params = {};
            var keyValues = parser.search.replace(/^\?/, '').split('&');
            keyValues.forEach(function (keyValue) {
                var pair = keyValue.split('=');
                params[pair[0]] = decodeURIComponent(pair[1] || '');
            });
            return params;
        })(),
        segments: parser.pathname.split('/').filter(function (seg) { return
seg !== ''; }),
        source: url,
        relative: parser.pathname + parser.search + parser.hash
    };
}

console.log(parse_URL('https://github.com/pubnub/python/search?utf8=%E2%9C%93
&q=python'));
```

JavaScript Validation

1. **Check if a value is boolean:**

```
function isBoolean(value) {
    return typeof value === 'boolean';
}
```

```
// Example usage:
console.log(isBoolean(true));  // true
console.log(isBoolean(false)); // true
console.log(isBoolean('true')); // false
```

2. **Check if a value is an Error object:**

```
function isError(value) {
    return value instanceof Error;
}

// Example usage:
console.log(isError(new Error('Sample error'))); // true
console.log(isError('Error message')); // false
```

3. **Check if a value is NaN:**

```
function isNaN(value) {
    return Number.isNaN(value);
}

// Example usage:
console.log(isNaN(NaN)); // true
console.log(isNaN('NaN')); // false
```

4. **Check if a value is null:**

```
function isNull(value) {
    return value === null;
}

// Example usage:
console.log(isNull(null)); // true
console.log(isNull(undefined)); // false
```

5. **Check if a value is a number:**

```
function isNumber(value) {
    return typeof value === 'number' && !Number.isNaN(value);
}

// Example usage:
console.log(isNumber(42)); // true
console.log(isNumber('42')); // false
```

6. **Check if a value is an object (excluding null):**

```
function isObject(value) {
    return typeof value === 'object' && value !== null;
}

// Example usage:
```

```
console.log(isObject({})); // true
console.log(isObject(null)); // false
```

7. **Check if a value is a plain JSON object (not an instance of a class):**

```
function isPlainJSONObject(value) {
    return typeof value === 'object' && value !== null && value.constructor
=== Object;
}

// Example usage:
console.log(isPlainJSONObject({})); // true
console.log(isPlainJSONObject(new Date())); // false
```

8. **Check if a value is a RegExp object:**

```
function isRegExp(value) {
    return value instanceof RegExp;
}

// Example usage:
console.log(isRegExp(/test/)); // true
console.log(isRegExp('/test/')); // false
```

9. **Check if a value is a character (string of length 1):**

```
function isChar(value) {
    return typeof value === 'string' && value.length === 1;
}

// Example usage:
console.log(isChar('a')); // true
console.log(isChar('abc')); // false
```

10. **Check if two values have the same type:**

```
function areSameType(value1, value2) {
    return typeof value1 === typeof value2;
}

// Example usage:
console.log(areSameType(42, '42')); // false (number vs string)
console.log(areSameType({}, [])); // false (object vs array)
console.log(areSameType(true, false)); // true (both are boolean)
```

Imp Questions:

The Movie Database

```
// Object to store information about a movie
let favoriteMovie = {
    title: "Puff the Magic Dragon",
    duration: 30,
    stars: ["Puff", "Jackie", "Living Sneezes"]
};

// Function to print movie information
function printMovie(movie) {
    console.log(`${movie.title} lasts for ${movie.duration} minutes. Stars:
${movie.stars.join(', ')}.`);
}

printMovie(favoriteMovie);
```

The Reading List

```
// Array of book objects
let books = [
    {
        title: "The Hobbit",
        author: "J.R.R. Tolkien",
        alreadyRead: true
    },
    {
        title: "The Lord of the Rings",
        author: "J.R.R. Tolkien",
        alreadyRead: false
    }
];

// Iterating through the array and logging book information
books.forEach(book => {
    console.log(`${book.title} by ${book.author}`);
    if (book.alreadyRead) {
        console.log(`You already read "${book.title}" by ${book.author}`);
    } else {
        console.log(`You still need to read "${book.title}" by
${book.author}`);
    }
});
```

The Recipe Card

```
// Object to hold recipe information
let favoriteRecipe = {
    title: "Mole",
    servings: 2,
    ingredients: ["cinnamon", "cumin", "cocoa"]
};

// Logging recipe information
console.log(favoriteRecipe.title);
console.log(`Serves: ${favoriteRecipe.servings}`);
console.log("Ingredients:");
```

```
favoriteRecipe.ingredients.forEach(ingredient => {
    console.log(ingredient);
});
```

The Fortune Teller

```
// Function to tell a fortune
function tellFortune(numChildren, partnerName, geoLocation, jobTitle) {
    console.log(`You will be a ${jobTitle} in ${geoLocation}, and married to
${partnerName} with ${numChildren} kids.`);
}

// Calling the function with different values
tellFortune(2, "Emma", "New York", "web developer");
tellFortune(3, "John", "California", "teacher");
tellFortune(0, "Sophia", "Florida", "pilot");
```

The Age Calculator

```
// Function to calculate age
function calculateAge(birthYear, currentYear) {
    let age1 = currentYear - birthYear;
    let age2 = age1 - 1;
    console.log(`You are either ${age1} or ${age2}`);
}

// Getting the current year
let currentYear = new Date().getFullYear();

// Calling the function with different values
calculateAge(1990, currentYear);
calculateAge(1985, currentYear);
calculateAge(2000, currentYear);
```

The Lifetime Supply Calculator

```
// Function to calculate lifetime supply
function calculateSupply(age, amountPerDay) {
    const maxAge = 80;
    let totalNeeded = Math.ceil((maxAge - age) * 365 * amountPerDay);
    console.log(`You will need ${totalNeeded} to last you until the ripe old
age of ${maxAge}`);
}

// Calling the function with different values
calculateSupply(30, 2);
calculateSupply(25, 3);
calculateSupply(40, 1.5);
```

The Geometrizer

```
// Function to calculate circumference of a circle
function calcCircumference(radius) {
    let circumference = 2 * Math.PI * radius;
    console.log(`The circumference is ${circumference.toFixed(2)}`);
}

// Function to calculate area of a circle
function calcArea(radius) {
    let area = Math.PI * radius * radius;
    console.log(`The area is ${area.toFixed(2)}`);
}

// Example usage
calcCircumference(5);
calcArea(5);
```

The Temperature Converter

```
// Function to convert Celsius to Fahrenheit
function celsiusToFahrenheit(celsius) {
    let fahrenheit = celsius * 9 / 5 + 32;
    console.log(`${celsius}°C is ${fahrenheit}°F`);
}

// Function to convert Fahrenheit to Celsius
function fahrenheitToCelsius(fahrenheit) {
    let celsius = (fahrenheit - 32) * 5 / 9;
    console.log(`${fahrenheit}°F is ${celsius}°C`);
}

// Example usage
celsiusToFahrenheit(30);
fahrenheitToCelsius(86);
```

Other Questions

Task 1: Basic Addition

Create an HTML page with JavaScript that prompts the user for two integers, adds them, and displays the result in an alert box.

HTML (`addition.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Addition</title>
  <script src="addition.js"></script>
</head>
<body>
  <!-- Content is not necessary for this task -->
</body>
```

```
</html>
```

JavaScript (`addition.js`):

```
let num1 = parseInt(prompt("Enter the first integer:"));
let num2 = parseInt(prompt("Enter the second integer:"));

// Add the numbers
let sum = num1 + num2;

// Display the result
alert(`The sum of ${num1} and ${num2} is ${sum}.`);
```

Task 2: Alphabetize a String

Write a JavaScript function that takes a string and returns the letters in alphabetical order.

```
javascript
Copy code
function alphabetizeString(str) {
  return str.split('').sort().join('');
}

// Example usage:
console.log(alphabetizeString("hello")); // Output: "ehllo"
```

Task 3: Extract Unique Characters

```
function uniqueCharacters(str) {
  let unique = '';
  for (let char of str) {
    if (unique.indexOf(char) === -1) {
      unique += char;
    }
  }
  return unique;
}

// Example usage:
console.log(uniqueCharacters("thequickbrownfoxjumpsoverthelazydog")); //
Output: "thequickbrownfxjmpsvlazydg"
```

Task 4: Convert Local Time to UTC and Display

HTML (`time.html`):

```
html
Copy code
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Time Conversion</title>
  <script src="time.js"></script>
</head>
<body>
  <div id="timeDisplay"></div>
  <button onclick="convertToLocalTime()">Convert to UTC</button>
</body>
</html>
```

JavaScript (`time.js`):

```
// Function to convert local time to UTC
function convertToLocalTime() {
  let now = new Date();
  let utcString = now.toUTCString();
  alert(`UTC Time: ${utcString}`);
}

// Function to display current time
function displayCurrentTime() {
  let now = new Date();
  let timeDisplay = document.getElementById('timeDisplay');
  timeDisplay.innerHTML = now.toLocaleString();
}

// Timer function to update time every second
setInterval(displayCurrentTime, 1000);
```

Task 5: Change Image on Click

HTML (`image.html`):

```
html
Copy code
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Change Image on Click</title>
  <script>
    function changeImage() {
      document.getElementById('myImage').src = 'new_image.jpg';
    }
  </script>
</head>
<body>
  <img id="myImage" src="original_image.jpg" onclick="changeImage()">
</body>
</html>
```

Task 6: Change Background Color on Button Click

HTML (`color.html`):

```
html
Copy code
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Change Background Color</title>
  <script>
    function changeColor() {
      document.body.style.backgroundColor = 'blue';
    }
  </script>
</head>
<body>
  <button onclick="changeColor()">Change Background Color</button>
</body>
</html>
```

Task 7: Insert and Style Paragraphs

HTML (`paragraphs.html`):

```
html
Copy code
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Insert Paragraph</title>
  <script src="paragraphs.js"></script>
</head>
<body>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
  <button onclick="insertParagraph()">Insert Paragraph</button>
</body>
</html>
```

JavaScript (`paragraphs.js`):

```
javascript
Copy code
function insertParagraph() {
  let newParagraph = document.createElement('p');
  newParagraph.textContent = 'New Paragraph';
  newParagraph.style.color = 'red'; // Example style change

  let secondParagraph = document.getElementsByTagName('p')[1];
  secondParagraph.parentNode.insertBefore(newParagraph,
secondParagraph.nextSibling);
}

// Function to revert style
function revertStyle() {
  let newParagraph = document.getElementsByTagName('p')[2]; // Assuming it's
the third paragraph
  newParagraph.style.color = ''; // Revert to original style
}

// Example event listener to revert style
document.addEventListener('DOMContentLoaded', function() {
  let newParagraph = document.getElementsByTagName('p')[2]; // Assuming it's
the third paragraph
  newParagraph.addEventListener('mouseout', revertStyle);
});
```

Task 8: Create a Slide Show of Images

Create a basic HTML page that displays a slideshow of 5 images.

HTML (`slideshow.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Slideshow</title>
  <script src="slideshow.js"></script>
</head>
<body>
  <div id="slideshow">
    <img src="image1.jpg" alt="Slide 1">
  </div>
</body>
</html>
```

JavaScript (`slideshow.js`):

```
let images = ['image1.jpg', 'image2.jpg', 'image3.jpg', 'image4.jpg',
'image5.jpg'];
let currentImageIndex = 0;
let slideshowElement = document.getElementById('slideshow');

function nextSlide() {
  currentImageIndex = (currentImageIndex + 1) % images.length;
  slideshowElement.innerHTML = `<img src="${images[currentImageIndex]}"
alt="Slide ${currentImageIndex + 1}">`;
}

// Automatically advance the slideshow every 3 seconds
setInterval(nextSlide, 3000);
```

Task 9: Add Navigation Buttons to Slideshow

HTML (`slideshow_nav.html`):

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Slideshow with Navigation</title>
  <script src="slideshow_nav.js"></script>
</head>
<body>
  <div id="slideshow">
    <img id="slide" src="image1.jpg" alt="Slide 1">
    <button onclick="prevSlide()" id="prevButton" style="display:
none;">Previous</button>
    <button onclick="nextSlide()" id="nextButton">Next</button>
  </div>
</body>
</html>
```

JavaScript (`slideshow_nav.js`):

```
let images = ['image1.jpg', 'image2.jpg', 'image3.jpg', 'image4.jpg',
'image5.jpg'];
let currentImageIndex = 0;
let slideshowElement = document.getElementById('slideshow');
let prevButton = document.getElementById('prevButton');
let nextButton = document.getElementById('nextButton');

function showSlide(index) {
  slideshowElement.innerHTML = `<img id="slide" src="${images[index]}"
alt="Slide ${index + 1}">`;
  currentImageIndex = index;

  // Toggle visibility of navigation buttons based on current slide index
  if (currentImageIndex === 0) {
    prevButton.style.display = 'none';
    nextButton.style.display = 'block';
  } else if (currentImageIndex === images.length - 1) {
    prevButton.style.display = 'block';
    nextButton.style.display = 'none';
  } else {
    prevButton.style.display = 'block';
    nextButton.style.display = 'block';
  }
}

function prevSlide() {
  if (currentImageIndex > 0) {
    showSlide(currentImageIndex - 1);
  }
}

function nextSlide() {
  if (currentImageIndex < images.length - 1) {
    showSlide(currentImageIndex + 1);
  }
}

// Initial slide display
```

```
showSlide(currentImageIndex);
```

Task 10: Create Animation Using setTimeout()

HTML (`animation.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Animation</title>
  <style>
    #animated {
      width: 100px;
      height: 100px;
      background-color: red;
      position: relative;
      animation: move 2s infinite;
    }
    @keyframes move {
      0% { left: 0; }
      50% { left: 50%; }
      100% { left: 0; }
    }
  </style>
  <script src="animation.js"></script>
</head>
<body>
  <div id="animated"></div>
</body>
</html>
```

JavaScript (`animation.js`):

```
let animatedElement = document.getElementById('animated');
let animationTimer;

animatedElement.addEventListener('mouseover', stopAnimation);
animatedElement.addEventListener('mouseout', startAnimation);

function stopAnimation() {
  clearTimeout(animationTimer);
  animatedElement.style.animationPlayState = 'paused';
}

function startAnimation() {
  animatedElement.style.animationPlayState = 'running';
  animate();
}

function animate() {
  animationTimer = setTimeout(() => {
    animatedElement.style.left = '50%';
    setTimeout(() => {
      animatedElement.style.left = '0';
      animate();
    }, 1000); // Delay after reaching 50% position
  }, 1000); // Initial delay before animation starts
}

// Start animation initially
startAnimation();
```

Task 11: Display Browser Information, Monitor Resolution, and Page Last Modified Date

HTML (`browser_info.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Browser Information</title>
  <script src="browser_info.js"></script>
</head>
<body>
  <!-- Content is not necessary for this task -->
</body>
</html>
```

JavaScript (`browser_info.js`):

```
document.write('<h2>Browser Information:</h2>');
document.write('<p>User Agent: ' + navigator.userAgent + '</p>');

document.write('<h2>Monitor Resolution:</h2>');
document.write('<p>Width: ' + screen.width + 'px</p>');
document.write('<p>Height: ' + screen.height + 'px</p>');

document.write('<h2>Page Information:</h2>');
document.write('<p>Last Modified: ' + document.lastModified + '</p>');
```

Task 12: Change Image on Hover

HTML (`image_hover.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Image Hover</title>
  <script src="image_hover.js"></script>
</head>
<body>
  <img src="image1.jpg" id="image1" onmouseover="changeImage(this)"
onmouseout="revertImage(this)">
  <img src="image2.jpg" id="image2" onmouseover="changeImage(this)"
onmouseout="revertImage(this)">
```

```
</body>
</html>
```

JavaScript (`image_hover.js`):

```
function changeImage(img) {
  if (img.src.match(/image1\.jpg$/)) {
    img.src = 'image1_hover.jpg';
  } else if (img.src.match(/image2\.jpg$/)) {
    img.src = 'image2_hover.jpg';
  }
}

function revertImage(img) {
  if (img.src.match(/image1_hover\.jpg$/)) {
    img.src = 'image1.jpg';
  } else if (img.src.match(/image2_hover\.jpg$/)) {
    img.src = 'image2.jpg';
  }
}
```

Task 13: Create a Signup Form with Validation

HTML (`signup_form.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Signup Form</title>
  <script src="signup_form.js"></script>
</head>
<body>
  <form id="signupForm" onsubmit="return validateForm()">
    <label for="username">Username:</label>
    <input type="text" id="username" name="username" required><br><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>
    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required><br><br>
    <input type="submit" value="Submit">
  </form>

  <div id="message"></div>
</body>
</html>
```

JavaScript (`signup_form.js`):

```
function validateForm() {
  let username = document.getElementById('username').value.trim();
  let email = document.getElementById('email').value.trim();
  let password = document.getElementById('password').value.trim();
  // Simple validation example (you can expand this as needed)
  if (username === '' || email === '' || password === '') {
    document.getElementById('message').textContent = 'All fields are
required.';
    return false;
  }

  // Additional validation logic (e.g., email format, password strength) can
be added here

  return true;
}
```

Task 14: Track Visits Using Cookies

HTML (`visit_tracker.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Visit Tracker</title>
  <script src="visit_tracker.js"></script>
</head>
<body>
  <h2>Welcome!</h2>
  <p>You have visited this page <span id="visitCount">0</span> times.</p>
  <p>Last visit on: <span id="lastVisit">Never</span></p>
  <img src="image.jpg" alt="Image">
</body>
</html>
```

JavaScript (`visit_tracker.js`):

```
function setCookie(name, value, days) {
  let expires = '';
  if (days) {
    let date = new Date();
    date.setTime(date.getTime() + (days * 24 * 60 * 60 * 1000));
    expires = '; expires=' + date.toUTCString();
  }
  document.cookie = name + '=' + value + expires + '; path=/';
}

function getCookie(name) {
  let nameEQ = name + '=';
  let cookies = document.cookie.split(';');
  for (let cookie of cookies) {
    while (cookie.charAt(0) === ' ') cookie = cookie.substring(1,
cookie.length);
    if (cookie.indexOf(nameEQ) === 0) return cookie.substring(nameEQ.length,
cookie.length);
  }
  return null;
}
```

```
function trackVisits() {
  let visitCount = getCookie('visitCount');
  let lastVisit = getCookie('lastVisit');

  let currentDate = new Date();
  let currentVisit = currentDate.toLocaleString();

  if (visitCount) {
    visitCount = parseInt(visitCount) + 1;
    document.getElementById('visitCount').textContent = visitCount;
  } else {
    visitCount = 1;
  }

  setCookie('visitCount', visitCount, 365);
  setCookie('lastVisit', currentVisit, 365);

  if (lastVisit) {
    document.getElementById('lastVisit').textContent = lastVisit;
  } else {
    document.getElementById('lastVisit').textContent = 'Never';
  }
}

// Track visits on page load
trackVisits();
```

Task 15: Create a "Mad Libs" Game

HTML (`madlibs.html`):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Mad Libs Game</title>
  <script src="madlibs.js"></script>
</head>
<body>
  <h2>Mad Libs Game</h2>
  <button onclick="playMadLibs()">Play Mad Libs</button>
  <div id="story"></div>
</body>
</html>
```

JavaScript (`madlibs.js`):

```
function playMadLibs() {
  let name = prompt('Enter your name:');
  let word1 = prompt('Enter a noun:');
  let word2 = prompt('Enter an adjective:');
  let word3 = prompt('Enter a verb:');
  let word4 = prompt('Enter a place:');
  let word5 = prompt('Enter a color:');
  let story = `
    Once upon a time, ${name} found a ${word1} in the ${word4}.
    It was very ${word2} and ${word3} ${word5}.
    ${name} decided to take it home.
  `;

  let coloredStory = story.replace(new
RegExp(`${name}|${word1}|${word2}|${word3}|${word4}|${word5}`, 'gi'),
function(matched) {
    return `<span style="color: ${word5};">${matched}</span>`;
  });

  document.getElementById('story').innerHTML = coloredStory;
}
```