1. Write a function that prints the numbers from 1 to 100. But for multiples of three, print "Fizz" instead of the number, and for the multiples of five, print "Buzz". For numbers that are multiples of both three and five, print "FizzBuzz".

Code:

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);

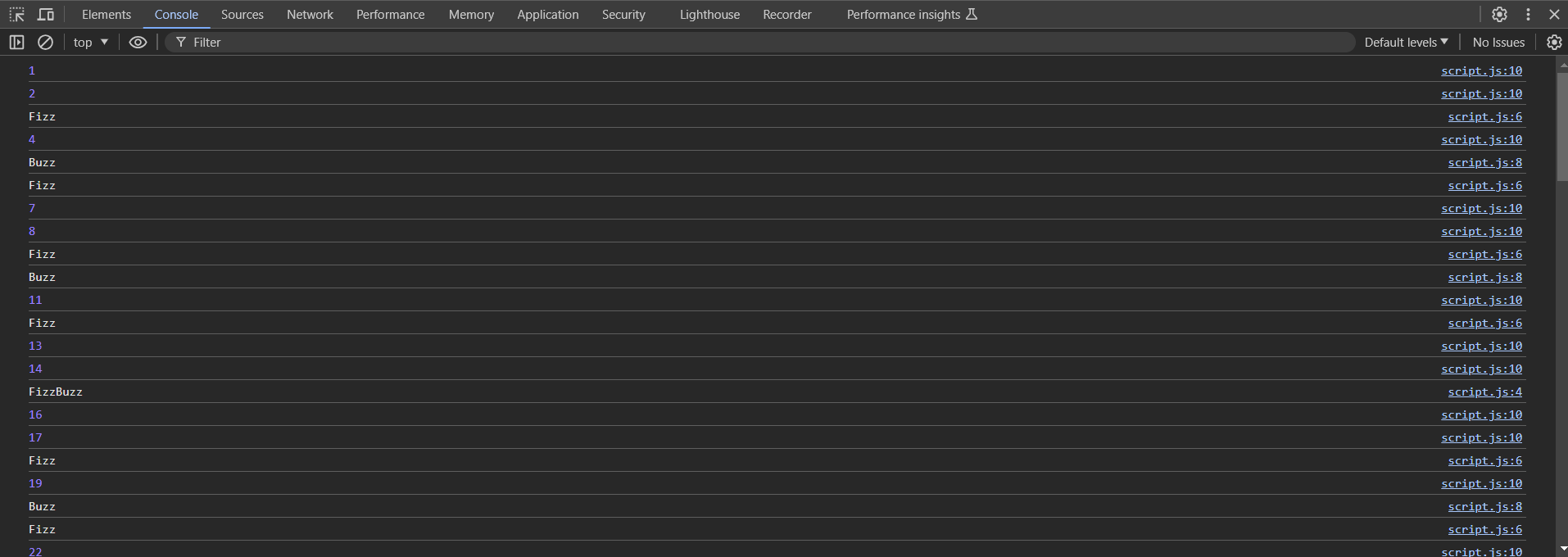
      }

    }

  }

  fizzBuzz();

output:



1. Write a function that takes a string input representing a simple arithmetic expression (only addition and subtraction) and returns the result.

Code:

function evaluateExpression(expression) {

    expression = expression.replace(/\s+/g, '');

    let numbers = expression.split(/[\+\-]/).map(Number);

    let operators = expression.split(/\d+/).filter(op => op);

    let result = numbers[0];

    for (let i = 0; i < operators.length; i++) {

      if (operators[i] === '+') {

        result += numbers[i + 1];

      } else if (operators[i] === '-') {

        result -= numbers[i + 1];

      }

    }

    return result;

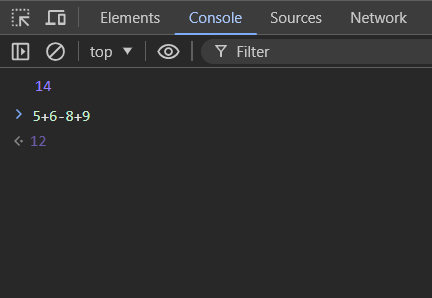
  }

  // Example usage

  let expression = "10 + 5 - 3 + 2";

  console.log(evaluateExpression(expression)); // Output: 14

output:



1. Write a function that takes a nested array and returns a flattened array.

Code:

function flattenArray(nestedArray) {

    let result = [];

    function flatten(element) {

      if (Array.isArray(element)) {

        element.forEach(flatten);

      } else {

        result.push(element);

      }

    }

    flatten(nestedArray);

    return result;

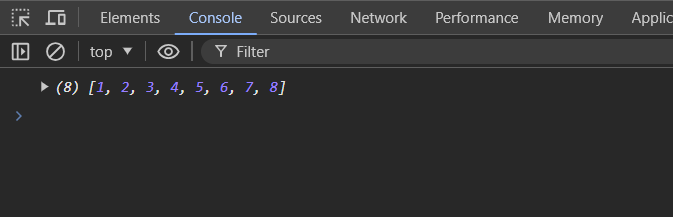
  }

  // Example usage

  let nestedArray = [1, [2, [3, 4], 5], 6, [7, 8]];

  console.log(flattenArray(nestedArray)); // Output: [1, 2, 3, 4, 5, 6, 7, 8]

output:



1. Write a function that checks if two given strings are anagrams of each other.

Code:

function areAnagrams(str1, str2) {

    // Helper function to clean and sort the string

    function cleanString(str) {

      return str.replace(/\s+/g, '').toLowerCase().split('').sort().join('');

    }

    // Clean and sort both strings

    let sortedStr1 = cleanString(str1);

    let sortedStr2 = cleanString(str2);

    // Compare the sorted strings

    return sortedStr1 === sortedStr2;

  }

  // Example usage

  let string1 = "listen";

  let string2 = "silent";

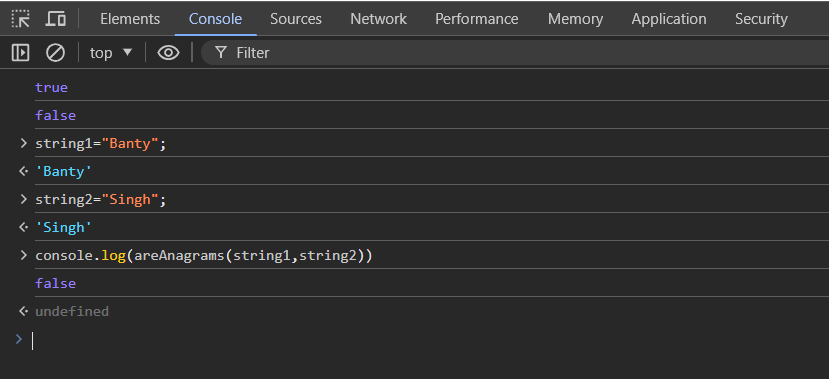
  console.log(areAnagrams(string1, string2)); // Output: true

  string1 = "hello";

  string2 = "world";

  console.log(areAnagrams(string1, string2)); // Output: false

output:



1. Write a function that takes an array and returns a new array with duplicates removed.

Code:

function removeDuplicates(array) {

    return [...new Set(array)];

  }

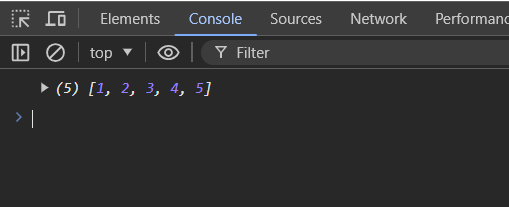
  // Example usage

  let arrayWithDuplicates = [1, 2, 3, 4, 3, 2, 1, 5];

  let arrayWithoutDuplicates = removeDuplicates(arrayWithDuplicates);

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

output:



1. Write a function that takes a string and capitalizes the first letter of each word in the string.

Code:

function capitalizeFirstLetterOfEachWord(str) {

    return str.split(' ').map(word => {

      return word.charAt(0).toUpperCase() + word.slice(1).toLowerCase();

    }).join(' ');

  }

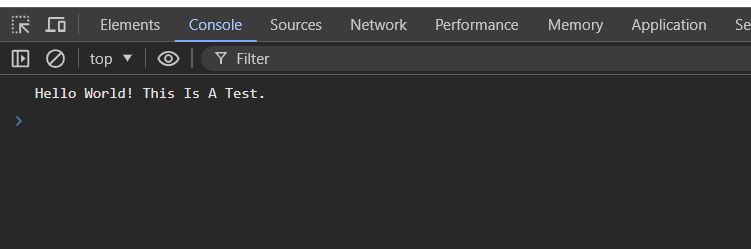
  // Example usage

  let sentence = "hello world! this is a test.";

  let capitalizedSentence = capitalizeFirstLetterOfEachWord(sentence);

  console.log(capitalizedSentence); // Output: "Hello World! This Is A Test."

Output:



1. Write a function that generates the first n numbers of the Fibonacci sequence.

Code:

function generateFibonacci(n) {

    if (n <= 0) {

      return [];

    }

    let fibonacciSequence = [0];

    if (n > 1) {

      fibonacciSequence.push(1);

    }

    for (let i = 2; i < n; i++) {

      let nextNumber = fibonacciSequence[i - 1] + fibonacciSequence[i - 2];

      fibonacciSequence.push(nextNumber);

    }

    return fibonacciSequence;

  }

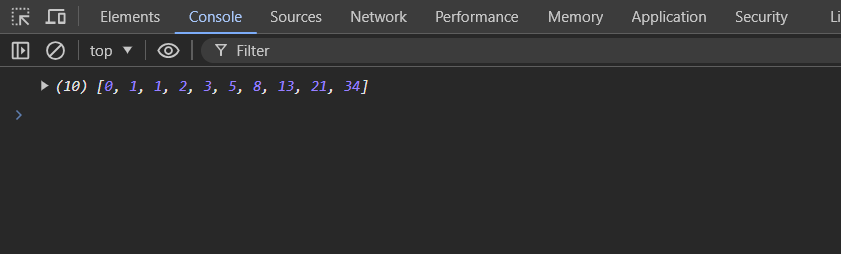
  // Example usage

  let n = 10;

  let fibonacciSequence = generateFibonacci(n);

  console.log(fibonacciSequence); // Output: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

output:



1. Implement a simple HashMap class with put, get, and remove methods.

Code:

class HashMap {

    constructor() {

      this.map = {};

    }

    // Method to add or update a key-value pair

    put(key, value) {

      this.map[key] = value;

    }

    // Method to retrieve the value associated with a key

    get(key) {

      return this.map.hasOwnProperty(key) ? this.map[key] : undefined;

    }

    // Method to remove a key-value pair

    remove(key) {

      if (this.map.hasOwnProperty(key)) {

        delete this.map[key];

      }

    }

  }

  // Example usage

  let myHashMap = new HashMap();

  myHashMap.put('name', 'Alice');

  console.log(myHashMap.get('name')); // Output: Alice

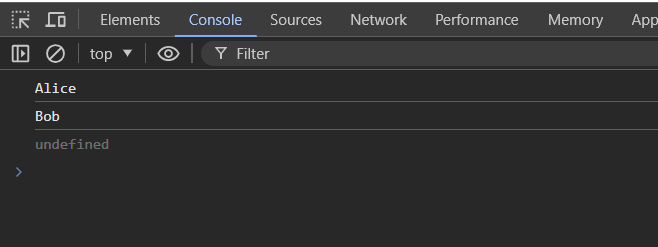
  myHashMap.put('name', 'Bob');

  console.log(myHashMap.get('name')); // Output: Bob

  myHashMap.remove('name');

  console.log(myHashMap.get('name')); // Output: undefined

output:



1. Write a function that filters out even numbers from an array.

Code:

function filterEvenNumbers(array) {

    return array.filter(number => number % 2 !== 0);

  }

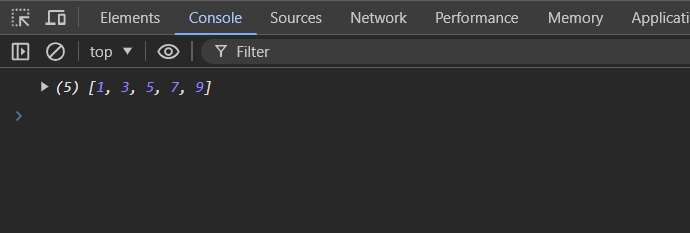
  // Example usage

  let numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

  let oddNumbers = filterEvenNumbers(numbers);

  console.log(oddNumbers); // Output: [1, 3, 5, 7, 9]

output:



1. Write a function that converts a given string to title case (capitalizing the first letter of each word).

Code:

function toTitleCase(str) {

    return str.split(' ').map(word => {

      return word.charAt(0).toUpperCase() + word.slice(1).toLowerCase();

    }).join(' ');

  }

  // Example usage

  let sentence = "hello world! this is a test.";

  let titleCasedSentence = toTitleCase(sentence);

  console.log(titleCasedSentence); // Output: "Hello World! This Is A Test."

Output:

