**1a. Print odd numbers in an array**

**in anonymous function**

**// Print odd numbers in an array in anonymous function**

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

**let printOddNumbers = function(num) {**

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

**if (num[i] % 2 !== 0) {**

**console.log(num[i]);**

**}**

**}**

**};**

**printOddNumbers(numbers);**

**1a. Output**

**[Running] node "d:\GUVI\JS\Practice\tempCodeRunnerFile.js"**

**1**

**3**

**5**

**7**

**9**

**[Done] exited with code=0 in 0.129 seconds**

**1a. Print odd numbers in an array**

**in IIFE**

**// Print odd numbers in an array in IIFE**

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

**(function(num) {**

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

**if (num[i] % 2 !== 0) {**

**console.log(num[i]);**

**}**

**}**

**})(numbers);**

**1a. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**1**

**3**

**5**

**7**

**9**

**[Done] exited with code=0 in 0.144 seconds**

**2a. Print odd numbers in an array**

**in arrow function**

**// Print odd numbers in an array in arrow function**

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

**let printOddNumbers = num => {**

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

**if (num[i] % 2 !== 0) {**

**console.log(num[i]);**

**}**

**}**

**};**

**printOddNumbers(numbers);**

**2a. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**1**

**3**

**5**

**7**

**9**

**[Done] exited with code=0 in 0.153 seconds**

**1b. Convert all the strings to title caps in a string array**

**in anonymous function**

**// Convert all the strings to title caps in a string array in anonymous function**

**let fruits = ["apple", "banana", "cherry", "date"];**

**let titleCaseConverter = function(cap) {**

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

**cap[i] = cap[i].charAt(0).toUpperCase() + cap[i].substring(1).toLowerCase();**

**}**

**};**

**titleCaseConverter(fruits);**

**console.log(fruits);**

**1b. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**[ 'Apple', 'Banana', 'Cherry', 'Date' ]**

**[Done] exited with code=0 in 0.12 seconds**

**1b. Convert all the strings to title caps in a string array**

**in IIFE**

**//Convert all the strings to title caps in a string array in IIFE**

**let fruits = ["apple", "banana", "cherry", "date"];**

**(function(cap) {**

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

**cap[i] = cap[i].charAt(0).toUpperCase() + cap[i].substring(1).toLowerCase();**

**}**

**})(fruits);**

**console.log(fruits);**

**1b. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**[ 'Apple', 'Banana', 'Cherry', 'Date' ]**

**[Done] exited with code=0 in 0.122 seconds**

**2b. Convert all the strings to title caps in a string array**

**in arrow function**

**//Convert all the strings to title caps in a string array in arrow function**

**let fruits = ["apple", "banana", "cherry", "date"];**

**let titleCaseConverter = cap => {**

**return cap.map(value => value.charAt(0).toUpperCase() + value.substring(1).toLowerCase());**

**};**

**let result = titleCaseConverter(fruits);**

**console.log(result);**

**1b. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**[ 'Apple', 'Banana', 'Cherry', 'Date' ]**

**[Done] exited with code=0 in 0.138 seconds**

**1c. Sum of all numbers in an array**

**in anonymous function**

**//Sum of all numbers in an array in anonymous function**

**let numbersArray = [1, 2, 3, 4, 5];**

**let sumOfNumbers = function(num) {**

**let sum = 0;**

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

**sum += num[i];**

**}**

**return sum;**

**};**

**let result = sumOfNumbers(numbersArray);**

**console.log("Sum of numbers:", result);**

**1c. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Sum of numbers: 15**

**[Done] exited with code=0 in 0.13 seconds**

**1c. Sum of all numbers in an array**

**in IIFE**

**//Sum of all numbers in an array in IIFE**

**let numbersArray = [1, 2, 3, 4, 5];**

**let sumOfNumbers = ((num) => {**

**return num.reduce((sum, num) => sum + num, 0);**

**})(numbersArray);**

**console.log("Sum of numbers:", sumOfNumbers);**

**1c. Output**

**[Running] node "d:\GUVI\JS\Practice\tempCodeRunnerFile.js"**

**Sum of numbers: 15**

**[Done] exited with code=0 in 0.117 seconds**

**2c. Sum of all numbers in an array**

**in arrow function**

**//Sum of all numbers in an array in arrow function**

**let numbersArray = [1, 2, 3, 4, 5];**

**let sumOfNumbers = (num) => num.reduce((sum, num) => sum + num, 0);**

**let result = sumOfNumbers(numbersArray);**

**console.log("Sum of numbers:", result);**

**1c. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Sum of numbers: 21**

**[Done] exited with code=0 in 0.122 seconds**

**1d. Return all the prime numbers in an array**

**in anonymous function**

**//Return all the prime numbers in an array in anonymous function**

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

**let findPrimeNumbers = function(value) {**

**let isPrime = function(num) {**

**if (num < 2) return false;**

**for (let i = 2; i <= Math.sqrt(num); i++) {**

**if (num % i === 0) return false;**

**}**

**return true;**

**};**

**return value.filter(num => isPrime(num));**

**};**

**let result = findPrimeNumbers(numbersArray);**

**console.log("Prime numbers:", result);**

**1d. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Prime numbers: [ 2, 3, 5, 7 ]**

**[Done] exited with code=0 in 0.128 seconds**

**1d. Return all the prime numbers in an array**

**in IIFE**

**//Return all the prime numbers in an array in IIFE**

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

**let primeNumbers = ((value) => {**

**let isPrime = (num) => {**

**if (num < 2) return false;**

**for (let i = 2; i <= Math.sqrt(num); i++) {**

**if (num % i === 0) return false;**

**}**

**return true;**

**};**

**return value.filter(num => isPrime(num));**

**})(numbersArray);**

**console.log("Prime numbers:", primeNumbers);**

**1d. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Prime numbers: [ 2, 3, 5, 7 ]**

**[Done] exited with code=0 in 0.148 seconds**

**2d. Return all the prime numbers in an array**

**in arrow function**

**//Return all the prime numbers in an array in arrow function**

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

**let primeNumbers = (value) => {**

**let isPrime = (num) => {**

**if (num < 2) return false;**

**for (let i = 2; i <= Math.sqrt(num); i++) {**

**if (num % i === 0) return false;**

**}**

**return true;**

**};**

**return value.filter(num => isPrime(num));**

**};**

**let result = primeNumbers(numbersArray);**

**console.log("Prime numbers:", result);**

**1d. Output**

**[Running] node "d:\GUVI\JS\Practice\tempCodeRunnerFile.js"**

**Prime numbers: [ 2, 3, 5, 7 ]**

**[Done] exited with code=0 in 0.133 seconds**

**1e. Return all the palindromes in an array**

**in anonymous function**

**//Return all the palindromes in an array in anonymous function**

**let words = ["level", "hello", "radar", "world", "civic"];**

**let findPalindromes = function(num) {**

**let isPalindrome = function(value) {**

**return value === value.split('').reverse().join('');**

**};**

**return num.filter(word => isPalindrome(word));**

**};**

**let result = findPalindromes(words);**

**console.log("Palindromes:", result);**

**1e. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Palindromes: [ 'level', 'radar', 'civic' ]**

**[Done] exited with code=0 in 0.141 seconds**

**1e. Return all the palindromes in an array**

**in IIFE**

**//Return all the palindromes in an array in IIFE**

**let words = ["level", "hello", "radar", "world", "civic"];**

**let palindromes = (function(value) {**

**let isPalindrome = function(str) {**

**return str === str.split('').reverse().join('');**

**};**

**return value.filter(word => isPalindrome(word));**

**})(words);**

**console.log("Palindromes:", palindromes);**

**1e. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Palindromes: [ 'level', 'radar', 'civic' ]**

**[Done] exited with code=0 in 0.122 seconds**

**2e. Return all the palindromes in an array**

**in arrow function**

**//Return all the palindromes in an array in arrow function**

**let words = ["level", "hello", "radar", "world", "civic"];**

**let findPalindromes = (value) => {**

**let isPalindrome = (str) => str === str.split('').reverse().join('');**

**return value.filter(word => isPalindrome(word));**

**};**

**let result = findPalindromes(words);**

**console.log("Palindromes:", result);**

**1e. Output**

**[Running] node "d:\GUVI\JS\Practice\tempCodeRunnerFile.js"**

**Palindromes: [ 'level', 'radar', 'civic' ]**

**[Done] exited with code=0 in 0.12 seconds**

**1f. Return median of two sorted arrays of the same size.**

**in anonymous function**

**//Return median of two sorted arrays of the same size in anonymous function**

**let numbers1 = [1, 3, 5, 7, 9];**

**let numbers2 = [2, 4, 6, 8, 10];**

**let findMedian = function(num1, num2) {**

**let mergedValue = num1.concat(num2).sort((a, b) => a - b);**

**let length = mergedValue.length;**

**if (length % 2 === 0) {**

**let mid1 = mergedValue[length / 2 - 1];**

**let mid2 = mergedValue[length / 2];**

**return (mid1 + mid2) / 2;**

**} else {**

**return mergedValue[Math.floor(length / 2)];**

**}**

**};**

**1f. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Median: 5.5**

**[Done] exited with code=0 in 0.124 seconds**

**1f. Return median of two sorted arrays of the same size.**

**in IIFE**

**//Return median of two sorted arrays of the same size in IIFE**

**let numbers1 = [1, 3, 5, 7, 9];**

**let numbers2 = [2, 4, 6, 8, 10];**

**let median = ((num1, num2) => {**

**let length = num1.length;**

**let medianIndex = Math.floor((2 \* length - 1) / 2);**

**let mergedValue = [];**

**for (let i = Math.max(0, medianIndex - length + 1); i <= Math.min(medianIndex, length - 1); i++) {**

**mergedValue.push(num1[i]);**

**mergedValue.push(num2[medianIndex - i]);**

**}**

**return mergedValue.reduce((sum, num) => sum + num) / mergedValue.length;**

**})(numbers1, numbers2);**

**console.log("Median:", median);**

**1f. Output**

**[Running] node "d:\GUVI\JS\Practice\tempCodeRunnerFile.js"**

**Median: 5.5**

**[Done] exited with code=0 in 0.107 seconds**

**1g. Remove duplicates from an array**

**in anonymous function**

**//Remove duplicates from an array in anonymous function**

**let withDuplicates = [1, 2, 1, 3, 4, 2, 5, 3, 1, 6];**

**let removeDuplicates = function(num) {**

**let uniqueValue = [];**

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

**if (uniqueValue.indexOf(num[i]) === -1) {**

**uniqueValue.push(num[i]);**

**}**

**}**

**return uniqueValue;**

**};**

**let result = removeDuplicates(withDuplicates);**

**console.log("Without duplicates:", result);**

**1g. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Without duplicates: [ 1, 2, 3, 4, 5, 6 ]**

**[Done] exited with code=0 in 0.122 seconds**

**1g. Remove duplicates from an array**

**in IIFE**

**//Remove duplicates from an array in IIFE**

**let withDuplicates = [1, 2, 1, 3, 4, 2, 5, 3, 1, 6];**

**let uniqueValue = ((num) => {**

**return num.filter((value, index, self) => self.indexOf(value) === index);**

**})(withDuplicates);**

**console.log("Without duplicates:", uniqueValue);**

**1g. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Without duplicates: [ 1, 2, 3, 4, 5, 6 ]**

**[Done] exited with code=0 in 0.148 seconds**

**1h. Rotate an array by k times**

**in anonymous function**

**//Rotate an array by k times in anonymous function**

**let arrayToRotate = [1, 2, 3, 4, 5];**

**let k = 3;**

**let rotateArray = function(num, k) {**

**for (let i = 0; i < k; i++) {**

**let temp = num.shift();**

**num.push(temp);**

**}**

**return num;**

**};**

**let rotatedArray = rotateArray(arrayToRotate, k);**

**console.log("Rotated Array:", rotatedArray);**

**1h. Output**

**[Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Rotated Array: [ 4, 5, 1, 2, 3 ]**

**[Done] exited with code=0 in 0.131 seconds**

**1h. Rotate an array by k times**

**in IIFE**

**//Rotate an array by k times in IIFE**

**let arrayToRotate = [1, 2, 3, 4, 5];**

**let k = 2;**

**let rotatedArray = ((num, rotations) => {**

**for (let i = 0; i < rotations; i++) {**

**let temp = num.shift();**

**num.push(temp);**

**}**

**return num;**

**})(arrayToRotate, k);**

**console.log("Rotated Array:", rotatedArray);**

**1h. Output**

**Running] node "d:\GUVI\JS\Practice\Day 4 task.js"**

**Rotated Array: [ 3, 4, 5, 1, 2 ]**

**[Done] exited with code=0 in 0.145 seconds**