1. Basic Calculator

# Functions for calculator operations

def add(a, b):

return a + b

def subtract(a, b):

return a - b

def multiply(a, b):

return a \* b

def divide(a, b):

if b == 0:

return "Error: Division by zero is not allowed."

return a / b

# Main program

def calculator():

try:

num1 = float(input("Enter the first number: "))

num2 = float(input("Enter the second number: "))

operator = input("Enter an operator (+, -, \*, /): ")

if operator == "+":

result = add(num1, num2)

elif operator == "-":

result = subtract(num1, num2)

elif operator == "\*":

result = multiply(num1, num2)

elif operator == "/":

result = divide(num1, num2)

else:

return "Invalid operator. Please use +, -, \*, or /."

return f"The result is: {result}"

except ValueError:

return "Invalid input. Please enter numeric values."

print(calculator())

3. Palindrome

function isPalindrome(str) {

const reversedStr = str.split('').reverse().join('');

return str === reversedStr;

}

function main() {

const inputStr = prompt("Enter a string:");

const isPalindromeResult = isPalindrome(inputStr);

if (isPalindromeResult) {

alert(`${inputStr} is a palindrome!`);

} else {

alert(`${inputStr} is not a palindrome.`);

}

}

main();

2 Temprature converter

function convertTemperature(temperature, scale) {

if (scale.toUpperCase() === 'C') {

return (temperature - 32) \* 5 / 9;

} else if (scale.toUpperCase() === 'F') {

return (temperature \* 9 / 5) + 32;

} else {

// Handle invalid input gracefully, perhaps by returning NaN or a specific error message

return NaN; // Or you can throw a custom error here

}

}

function main() {

const temperature = parseFloat(prompt("Enter the temperature:"));

const scale = prompt("Enter the scale to convert to (C/F):");

const convertedTemperature = convertTemperature(temperature, scale);

if (isNaN(convertedTemperature)) {

alert("Invalid input. Please enter a valid temperature and scale.");

} else {

alert(`The converted temperature is: ${convertedTemperature.toFixed(2)} ${scale.toUpperCase()}`);

}

}

main();

4 . factorial

function factorial(num) {

if (num < 0) {

return "Factorial is not defined for negative numbers.";

} else if (num === 0 || num === 1) {

return 1;

} else {

let result = 1;

for (let i = 2; i <= num; i++) {

result \*= i;

}

return result;

}

}

function main() {

const number = parseInt(prompt("Enter a non-negative integer:"));

const factorialResult = factorial(number);

if (typeof factorialResult === 'string') {

alert(factorialResult);

} else {

alert(`The factorial of ${number} is ${factorialResult}`);

}

}

main();

5. Array Sum

function sumArray(array) {

let sum = 0;

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

sum += array[i];

}

return sum;

}

function main() {

const myArray = [1, 2, 3, 4, 5];

const arraySum = sumArray(myArray);

console.log("The sum of the array elements is:", arraySum);

}

main();

6. Fibonacci Sequence Function

function fibonacci(n) {

const fibSequence = [0, 1];

for (let i = 2; i < n; i++) {

fibSequence[i] = fibSequence[i - 1] + fibSequence[i - 2];

}

return fibSequence;

}

function main() {

const n = 10; // Number of terms

const fibSeq = fibonacci(n);

console.log("Fibonacci sequence:", fibSeq);

}

main();

7. Find Maximum in an Array

function findMax(array) {

if (array.length === 0) {

return undefined; // Handle empty array case

}

let max = array[0];

for (let i = 1; i < array.length; i++) {

if (array[i] > max) {

max = array[i];

}

}

return max;

}

function main() {

const myArray = [23, 45, 12, 90, 56];

const maxNumber = findMax(myArray);

console.log("The maximum number is:", maxNumber);

}

main();

8. Reverse a String Function

function reverseString(str) {

let reversedStr = "";

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

reversedStr += str[i];

}

return reversedStr;

}

function main() {

const inputStr = "hello world";

const reversedStr = reverseString(inputStr);

console.log("Reversed string:", reversedStr);

}

main();

9. Count Vowels in a String

function countVowels(str) {

const vowels = "aeiouAEIOU";

let count = 0;

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

if (vowels.includes(str[i])) {

count++;

}

}

return count;

}

function main() {

const inputStr = "hello world";

const vowelCount = countVowels(inputStr);

console.log("Number of vowels:", vowelCount);

}

main();

10. Get Random Number in Range

function getRandomNumber(min, max) {

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

function main() {

const minNum = 1;

const maxNum = 10;

const randomNum = getRandomNumber(minNum, maxNum);

console.log("Random number:", randomNum);

}

main();

11. Array Filter Function

function filterArray(array, condition) {

const filteredArray = [];

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

if (condition(array[i])) {

filteredArray.push(array[i]);

}

}

return filteredArray;

}

function main() {

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

const evenNumbers = filterArray(numbers, num => num % 2 === 0);

console.log("Even numbers:", evenNumbers);

}

main();

12. Concatenate Arrays

function concatArrays(arr1, arr2) {

return arr1.concat(arr2);

}

function main() {

const array1 = [1, 2, 3];

const array2 = [4, 5, 6];

const combinedArray = concatArrays(array1, array2);

console.log("Combined array:", combinedArray);

}

main();

**13 Remove Duplicates from an Array**

function removeDuplicates(array) {

const uniqueArray = [];

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

if (!uniqueArray.includes(array[i])) {

uniqueArray.push(array[i]);

}

}

return uniqueArray;

}

function main() {

const numbers = [1, 2, 3, 2, 1, 4, 5, 4];

const uniqueNumbers = removeDuplicates(numbers);

console.log("Unique numbers:", uniqueNumbers);

}

main()

14. Check Even or Odd

function isEven(num) {

return num % 2 === 0;

}

function main() {

const number = parseInt(prompt("Enter a number:"));

if (isEven(number)) {

console.log(`${number} is even.`);

} else {

console.log(`${number} is odd.`);

}

}

main();

15. Greet User Function

function greetUser(name) {

console.log("Hello, " + name + "!");

}

function main() {

const name = prompt("Enter your name:");

greetUser(name);

}

main();

14. Array Methods (map, filter, reduce)

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

// Map: Square each number

const squaredNumbers = numbers.map(number => number \* number);

console.log("Squared numbers:", squaredNumbers);

// Filter: Even numbers

const evenNumbers = numbers.filter(number => number % 2 === 0);

console.log("Even numbers:", evenNumbers);

// Reduce: Sum of all numbers

const sum = numbers.reduce((accumulator, number) => accumulator + number, 0);

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

**Object Literal Enhancements**

const name = "Alice";

const age = 30;

const person = {

name,

age

};

console.log(person);

**Set and Map**

// Set

const mySet = new Set();

mySet.add(1);

mySet.add(2);

mySet.add(3);

mySet.add(2); // Duplicate, will not be added

console.log("Set:", mySet);

// Map

const myMap = new Map();

myMap.set("name", "Alice");

myMap.set("age", 30);

myMap.set("city", "New York");

for (const [key, value] of myMap) {

console.log(`${key}: ${value}`);

}

**Promises**

function wait(seconds) {

return new Promise(resolve => {

setTimeout(() => {

resolve(`Waited for ${seconds} seconds.`);

}, seconds \* 1000);

});

}

wait(2)

.then(message => console.log(message));

**Classes**

class Rectangle {

constructor(width, height) {

this.width = width;

this.height = height;

console.log(`Rectangle created with width: ${width} and height: ${height}`);

this.calculateArea();

}

calculateArea() {

const area = this.width \* this.height;

console.log(`Area of the rectangle: ${area}`);

}

}

const myRectangle = new Rectangle(5, 10);

**Rest Parameters**

function sumAll(...numbers) {

let sum = 0;

for (const number of numbers) {

sum += number;

}

return sum;

}

console.log(sumAll(1, 2, 3)); // Output: 6

console.log(sumAll(10, 20, 30, 40)); // Output: 100

**Destructuring Assignment**

const person = {

name: "Alice",

age: 30,

city: "New York"

};

const { name, age } = person;

console.log(name); // Output: Alice

console.log(age); // Output: 30

**Spread Operator**

const array1 = [1, 2, 3];

const array2 = [4, 5, 6];

const combinedArray = [...array1, ...array2];

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

**Using let and const**

let x = 10;

console.log(x); // Output: 10

x = 20;

console.log(x); // Output: 20

const y = 30;

console.log(y); // Output: 30

// Uncomment the following line to see an error:

// y = 40;

// Traditional function

**Arrow Functions**

function sum(a, b) {

return a + b;

}

// Arrow function

const sumArrow = (a, b) => a + b;

console.log(sum(2, 3)); // Output: 5

console.log(sumArrow(2, 3)); // Output: 5

**Template Literals**

const firstName = "Alice";

const lastName = "Johnson";

const age = 30;

const greeting = `Hello, ${firstName} ${lastName}! You are ${age} years old.`;

console.log(greeting);

**Default Parameters**

function greet(name, greeting = "Hello") {

console.log(`${greeting}, ${name}!`);

}

greet("Bob"); // Output: Hello, Bob!

greet("Alice", "Hi"); // Output: Hi, Alice!

**Promises and Asynchronous JavaScript**

function simulateAsyncOperation(delay) {

return new Promise((resolve, reject) => {

setTimeout(() => {

resolve(`Task completed after ${delay} seconds.`);

}, delay \* 1000);

});

}

simulateAsyncOperation(3)

.then(result => console.log(result))

.catch(error => console.error(error));

**Higher-Order Functions and Callbacks**

function mapArray(array, callback) {

const newArray = [];

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

newArray.push(callback(array[i]));

}

return newArray;

}

const numbers = [1, 2, 3, 4, 5];

// Square each number

const squaredNumbers = mapArray(numbers, number => number \* number);

console.log(squaredNumbers); // Output: [1, 4, 9, 16, 25]

// Double each number

const doubledNumbers = mapArray(numbers, number => number \* 2);

console.log(doubledNumbers); // Output: [2, 4, 6, 8, 10]