*Assignment*

Que.01 Create an arrow function called square that takes a number as an argument and returns its square. Use the arrow function to calculate the square of a given number and display the result.

Ans. *// Arrow function to calculate the square of a number*

const square **=** (**number**) **=>** {

**return** number **\*** number

}

*// Calculate the square of a given number*

const inputNumber **=** 7**;**

const result **=** square(inputNumber)

*// Display the result*

console**.**log(`The square of ${inputNumber} is ${result}`)**;**

Que.02 Create a JavaScript function called generateGreeting that takes a name as an argument and returns a personalized greeting message. Use this function to greet three different people.

Ans. *// Function to generate a personalized greeting*

function generateGreeting(**name**){

**return** `Hello, ${name}! Welcome to our website.`**;**

}

*// Greet three different peoples*

const person1 **=** "Alice"**;**

const person2 **=** "Bob"**;**

const person3 **=** "Charlie"**;**

const greeting1 **=** generateGreeting(person1)**;**

const greeting2 **=** generateGreeting(person2)**;**

const greeting3 **=** generateGreeting(person3)**;**

*// Display the greetings*

console**.**log(greeting1)**;**

console**.**log(greeting2)**;**

console**.**log(greeting3)**;**

Que.03 Create an IIFE (Immediately Invoked Function Expression) that calculates the square of a number and immediately displays the result.

Ans. (function() {

// Define the number

var number = 5;

// Calculate the square

var square = number \* number;

// Display the result

console.log("The square of", number, "is", square);

})();

Que.04 Write a JavaScript function called calculateTax that takes an income as an argument and returns the amount of tax to be paid. Use a closure to handle different tax rates based on income ranges. Test the function with various incomes.

Ans. function calculateTax() {

// Define tax rates based on income ranges using closure

var taxRates = [

{ range: 10000, rate: 0.1 },

{ range: 30000, rate: 0.15 },

{ range: 70000, rate: 0.2 },

{ range: Infinity, rate: 0.25 }

];

// Return a function that calculates tax based on income

return function(income) {

var tax = 0;

for (var i = 0; i < taxRates.length; i++) {

if (income <= taxRates[i].range) {

tax += income \* taxRates[i].rate;

break;

} else {

tax += taxRates[i].range \* taxRates[i].rate;

income -= taxRates[i].range;

}

}

return tax;

};

}

// Test the function with various incomes

var calculateTaxFunction = calculateTax();

console.log("Tax for $5,000 income:", calculateTaxFunction(5000));

console.log("Tax for $20,000 income:", calculateTaxFunction(20000));

console.log("Tax for $50,000 income:", calculateTaxFunction(50000));

console.log("Tax for $100,000 income:", calculateTaxFunction(100000));

Que.05 Write a JavaScript function called factorial that calculates the factorial of a non-negative integer using recursion. Test the function with different inputs.

Ans. function f(n) {

if(n == 0) return 1;

const result = n \* f(n-1);

return result;

}

console.log(1); //output 1;

console.log(f(3)); //output 6;

console.log(f(5)); //output 120;

console.log(f(7)); //output 5040;

console.log(f(9)); //output 362880;

console.log(f(10)); //output 3628800;

console.log(f(11)); //output 39916800;

Que.06 Write a JavaScript function called curry that takes a function as an argument and returns a curried version of that function. The curried function should accept arguments one at a time and return a new function until all arguments are provided. Then, it should execute the original function with all arguments. Test the curry function with a function that adds two numbers.

Ans.

function curry(func) {

return function curried(...args) {

if (args.length >= func.length) {

// If all arguments are provided, execute the original function

return func(...args);

} else {

// If not all arguments are provided, return a new function that accepts the remaining arguments

return function(...moreArgs) {

return curried(...args, ...moreArgs);

};

}

};

}

// Test the curry function with a function that adds two numbers

function add(x, y) {

return x + y;

}

const curriedAdd = curry(add);

// Test the curried function with one argument at a time

const add5 = curriedAdd(5);

console.log(add5(3)); // Output: 8

// Alternatively, provide both arguments at once

console.log(curriedAdd(2, 4)); // Output: 6