**Java script tasks**

1. **Variables: Write a program that declares a variable age and assigns your age to it. Then, log the value of age to the console.**

var age = 10;

console.log(age)

1. **Functions: Create a function called calculateArea that takes two parameters length and width and returns the calculated area. Test the function by passing different values and logging the result.**

function calculateArea(len,wid){

return 2\*len\*wid

}

var area = calculateArea(10,10)

console.log(area)

1. **Type Coercion: Write a program that concatenates a number with a string and prints the result without explicitly converting the number to a string.**

var number = 43;

var strg = "apple"

console.log(number+strg)

1. **Data Types and Literals: Create variables to represent different data types (number, string, boolean, array, object) and log their types using the typeof operator.**

var number = 43;

var strg = "apple";

var boo = true;

var arr = [1,2,3];

var newobj = new Object();

console.log(typeof(number))

console.log(typeof(strg))

console.log(typeof(boo))

console.log(typeof(arr))

console.log(typeof(newobj))

It shows object eventhough we call typeof for an array variable. It is because,

Js treats array as a special kind of object only. But it is an array.

1. **Built-in Objects - Date: Write a program that gets the current date and time using the Date object and displays it in a human-readable format.**

var currentDate = new Date();

let year = currentDate.getFullYear();

let month = currentDate.getMonth(); //month is from (0-11 so +1 is done)

let day = currentDate.getDate();

console.log(day+"/"+month+1+"/"+year)

1. **Built-in Operators: Create a program that takes two numbers as input and performs arithmetic operations (addition, subtraction, multiplication, division) on them. Display the results.**

//parsefloat to convert the user input to number

var num1 = parseFloat(prompt("enter num1: "))

var num2 = parseFloat(prompt("enter num2: "))

console.log(num1+num2)

console.log(num1-num2)

console.log(num1\*num2)

console.log(num1/num2)

1. **String Manipulation: Write a function that takes a string as input and returns the string reversed.**

var str1 = "Hello";

function revstr(x){

var rev = '';

for(var i = str1.length - 1;i >= 0;i--){

rev += str1[i];

}

return rev;

}

var out = revstr(str1)

console.log(out)

1. **String to Array: Create a program that converts a given string into an array of characters and then joins it back into a string. Display the final string.**

var str1 = "Hello";

function arrstr(x){

var rev = [];

var str2 = '';

for(var i = 0;i < str1.length;i++){

rev.push(str1[i])

}

for(var j = 0;j<rev.length;j++){

str2+=rev[j]

}

return str2;

}

var out = arrstr(str1)

console.log(out)

1. **Flow Control Structures (if-else): Write a program that checks if a given number is even or odd and prints the result.**

var num = 23456;

if(num%2 == 0){

console.log("even")

}

else{

console.log("odd")

}

1. **Flow Control Structures (switch): Create a program that takes a day of the week as input (string or number) and outputs a corresponding message based on whether it's a weekday or weekend.**

var day = 1;

switch (day){

case 1:

case 2:

case 3:

case 4:

case 5:

console.log("It's a weekday :( ")

break;

case 6:

case 7:

console.log("It's a weekend :) ")

break;

default:

console.log("There are only seven days in a week from 1 to 7.")

}

1. **Prototype: Define a constructor function Book with properties title and author. Create a prototype method getInfo that returns a string containing book inform ation. Instantiate a Book object and call the getInfo method.**

function Book(title,author){

this.title = title

this.author = author

}

Book.prototype.getInfo = function(){

console.log("The book named "+this.title+" is written by "+this.author)

}

var book = new Book("After","Anna Todd")

book.getInfo()

1. **Event Handlers (addEventListener): Implement a button in HTML with an associated event listener in JavaScript. When the button is clicked, display an alert message.**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>eventlistener</title>

<style>

button{

color: white;

background-color: black;

border-style: none;

box-shadow: 2px 2px 10px rgb(141, 141, 141);

border-radius: 20px;

width: 80px;

height: 40px;

}

</style>

</head>

<body>

<button id="butn">Don't</button>

<script>

var button = document.getElementById("butn")

button.addEventListener("click",function(){

alert("I said don't")

})

</script>

</body>

</html>

1. **Functions with Variable Scope: Create a program with a function that declares a variable inside it and attempts to access it outside the function. Explain the result.**

var x = 5;

function test(){

var x = 10;

console.log(x)

}

test(); //prints 10 //local declaration

console.log(x) //output : 5 //global declaration

1. **Function Callbacks: Write a function square that takes an array of numbers and a callback function. The callback function should square each number in the array. Test it with different arrays.**

function addition(a,b,call){

this.a = a;

this.b = b;

call(a,b)

}

function sum(a,b){

console.log(a+b);

}

addition(3,5,sum)

1. **Asynchronous Operations (Promises): Implement a function that simulates an asynchronous operation (e.g., fetching data) using a Promise. Resolve the Promise after a delay and log a success message.**

function promise(){

return new Promise((resolve) => {

setTimeout(() => {

resolve('The work is completed')

}, 2000);

});

}

promise().then((data)=>{console.log('The problem is resolved ' + data);})

.catch((error)=> {console.log('The problem rejected ' + error);})

1. **Closures: Create a function counter that returns another function. The returned function, when called, increments a counter variable defined in the outer function and returns the updated count.**

function counter(){

let count = 0;

return function(){

++count;

console.log(count);

}

}

var c = counter();

c(); //output 1

c(); //output 2

1. **Higher-Order Functions: Write a program that uses higher-order functions (e.g., map, filter, reduce) on an array of numbers to perform various operations like doubling, filtering even numbers, or finding the sum.**

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

let marr = arr.map((e)=>e\*2);

console.log(marr);

let farr = arr.filter((e)=>e%2===0);

console.log(farr);

let rarr = arr.reduce((sum,e)=>sum+e,0);

console.log(rarr);

1. **Error Handling (try-catch): Implement a program that attempts to access an undefined variable within a try block and catches the error using a catch block.**

try{

console.log(a);

}

catch(error){

console.log("Error: ", error.message);

}

1. **Object-Oriented Programming: Create a class Car with properties make, model, and a method getDetails that returns the car's details. Instantiate a Car object and call the getDetails method.**

class Car{

constructor(make,model){

this.make = make;

this.model = model;

}

getDetails(){

console.log('Make: ' +this.make);

console.log('Model: ' + this.model);

}

}

var car = new Car('Toyata', 'Camry');

car.getDetails();