Task-1

Aim:

Variables and Data Types.

Description:

Declare a variable using var, let, and const. Assign different data types to each variable and print their values.

Source Code:

```
<!DOCTYPE html>

<html lang="en">

<head>

   <meta charset="UTF-8">

   <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

   <title>Document</title>

</head>

<body>

   <p id="var"></p>

   <p id="let"></p>

   <p id="const"></p>

   <script>

   // Using var

   var myVar = "Hello";

   document.getElementById("var").innerHTML=myVar;
```

```
    // Using let

    let myLet = 42;

    document.getElementById("let").innerHTML=myLet;

    // Using const

    const myConst = true;

    document.getElementById("const").innerHTML=myConst;

    </script>

</body>

</html>
```

Output:

Hello

42

true

Task-2

Aim:

Operators and Expressions.

Description:

Write a function that takes two numbers as arguments and returns their sum, difference, product, and quotient using arithmetic operators.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        function performArithmeticOperations(num1, num2) {
 var sum = num1 + num2;
 var difference = num1 - num2;
 var product = num1 * num2;
 var quotient = num1 / num2;

 return {
    sum: sum,
    difference: difference,
    product: product,
    quotient: quotient
 };
}

// Example usage
var num1 = 10;
var num2 = 5;

var result = performArithmeticOperations(num1, num2);


console.log("Sum:", result.sum);
console.log("Difference:", result.difference);
console.log("Product:", result.product);
console.log("Quotient:", result.quotient);

    </script>
</body>
</html>
```

Output:

Sum: – 15
Difference: – 5
Product: – 50
Quotient: – 2
Live reload enabled.

Task-3

Aim:

Control Flow.

Description:

Write a program that prompts the user to enter their age. Based on their age, display different messages:

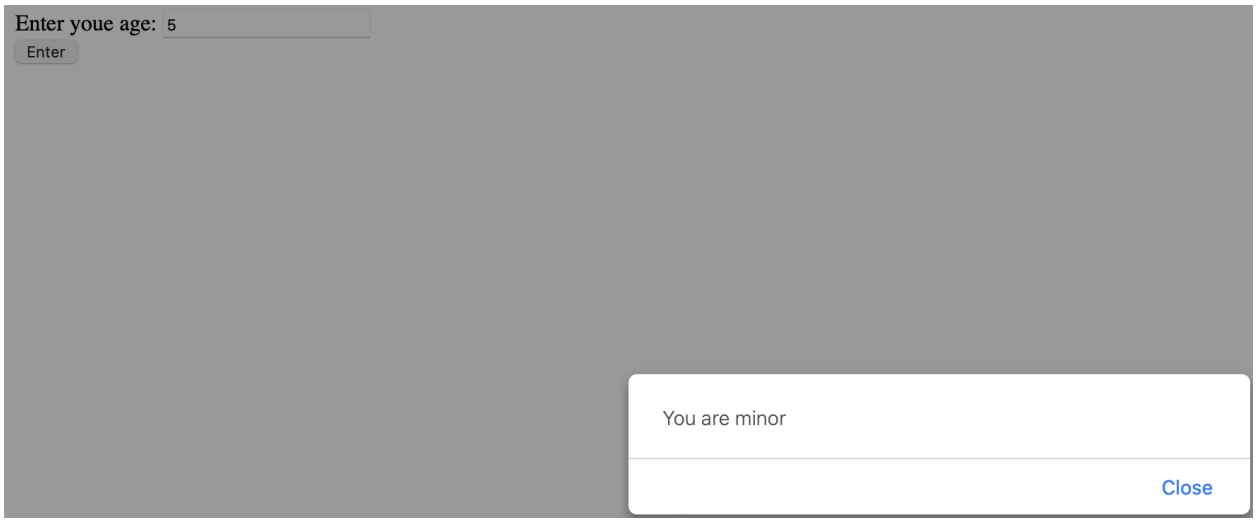
○ If the age is less than 18, display "You are a minor."
○ If the age is between 18 and 65, display "You are an adult."
○ If the age is 65 or older, display "You are a senior citizen."

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <div>
       <label>Enter youe age: </label>
       <input type="text" name="age" id="age">

       <br><input type="button" value="Enter" onclick="fun()">
```

```
        <label id="abc"></label>
    </div>
    <script>
        function fun(){

            let age=document.getElementById("age").value;

            if(age<18){
                alert("You are minor");
            }
            if(18<age<65){
                alert("you are adult");
            }
            if(age>65){
                alert("you are a senior citizen");
            }
        }
    </script>
</body>
</html>
```

Output:



Task-4

Aim:

Functions.

Description:

Write a function that takes an array of salary as an argument and returns the min/max salary in the array.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>min max</title>
</head>
<body>
   min:
   <p id="min"></p><hr>
   max:
   <p id="max"></p>

   <script>
       function findMinMaxSalary(salaries) {
 var minSalary = Math.min(...salaries);
 var maxSalary = Math.max(...salaries);

 return {
   minSalary: minSalary,
   maxSalary: maxSalary,
 };
}

// Example usage
var salaries = [50000, 60000, 45000, 70000, 55000];
var result = findMinMaxSalary(salaries);
```

```
document.getElementById("min").innerHTML=result.minSalary;
document.getElementById("max").innerHTML=result.maxSalary;
   </script>
</body>
</html>
```

Output:

> min:
>
> 45000
>
> ---
>
> max:
>
> 70000

Task-5

Aim:

Arrays and Objects.

Description:

Create an array of your favorite books. Write a function that takes the array as an argument and displays each book title on a separate line.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>fav book</title>
</head>
<body>
   <p id="book-list"></p>
```

```
   <script>

var favoriteBooks = [
 "To Kill a Mockingbird",
 "1984",
 "Harry Potter and the Sorcerer's Stone",
 "The Lord of the Rings",
 "Pride and Prejudice",
];
function displayBooks(books) {
 var bookList = document.getElementById("book-list");

 for (var i = 0; i < books.length; i++) {
   var bookTitle = document.createElement("li");
   bookTitle.textContent = books[i];
   bookList.appendChild(bookTitle);
 }
}
displayBooks(favoriteBooks);
   </script>
</body>
</html>
```

Output:

- To Kill a Mockingbird
- 1984
- Harry Potter and the Sorcerer's Stone
- The Lord of the Rings
- Pride and Prejudice

Task-6

Aim:

Scope and Hoisting.

Description:

Declare a variable inside a function and try to access it outside the function. Observe the scope behavior and explain the results. [var vs let vs const]

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <script>
       //var
       function myFunction() {
     var x = 10;
    }
   myFunction();
   console.log(x); // Output: ReferenceError: x is not defined

   //let
   function myVAR() {
   let y = 20;
   console.log(y); // Output: 20
   }
   myVAR();
   console.log(y); // Output: ReferenceError: y is not defined

   //const
   function con() {
   const z = 30;
   console.log(z); // Output: 30
   }
   con();
   console.log(z); // Output: ReferenceError: z is not defined
```

```
    </script>
</body>
</html>
```

Output:

ReferenceError: Can't find variable: x

Task-7

Aim:

DOM Manipulation.

Description:

Create an HTML page with a button. Write JavaScript code that adds an event listener to the button and changes its text when clicked.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>clicked</title>

    <style>

    </style>
</head>
<body>
    <button id="myButton">Click Me</button>

    <script>
        var button = document.getElementById("myButton");
```

```
        // Add event listener to the button
        button.addEventListener("click", function() {
        button.textContent = "Clicked!";
    });
    </script>
</body>
</html>
```

Output:

Task-8

Aim:

Error Handling.

Description:

Write a function that takes a number as an argument and throws an error if the number is negative. Handle the error and display a custom error message.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>error</title>
</head>
<body>
    <p id="error"></p>
    <script>
        function processNumber(num) {
```

```
 if (num < 0) {
   throw new Error("Number cannot be negative.");
 }
  return num * 2;
}

try {
 var result = processNumber(-5);
 console.log("Result:", result); // This line won't be executed
} catch (error) {
 document.getElementById("error").innerHTML=error.message;
}
   </script>
</body>
</html>
```

Output:

Number cannot be negative.

Task-9

Aim:

Asynchronous JavaScript.

Description:

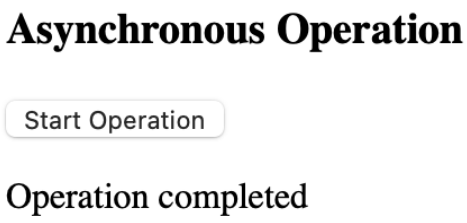
Write a function that uses setTimeout to simulate an asynchronous operation. Use a callback function to handle the result.

Source Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
```

```
</head>
<h3>Asynchronous Operation</h3>
 <button id="startButton">Start Operation</button>
 <p id="result"></p>
<body>
   <script>
function simulateAsyncOperation(callback) {
 setTimeout(function() {
   var result = "Operation completed";
   callback(result);
 }, 2000);
}
function handleResult(result) {
 var resultElement = document.getElementById("result");
 resultElement.textContent = result;
}

var startButton = document.getElementById("startButton");
startButton.addEventListener("click", function() {
 simulateAsyncOperation(handleResult);
});
   </script>
</body>
</html>
```

Output:



Learning Outcome:

1 : Understand various technologies and trends impacting single page web applications.