1.

var num1=prompt("enter the first number");

var num2=prompt("enter the second number");

num1=parseFloat(num1);

num2=parseFloat(num2);

var sum= num1+num2;

alert("the sum is"+num1+"and"+num2+"is:"+sum);

1b] const reverseNumber = num => {

  const reversed = parseFloat(num.toString().split('').reverse().join('')) \* Math.sign(num);

  return reversed;

}

// Example usage

const num = 12243;

const reversedNum = reverseNumber(num);

console.log(reversedNum); // Output: 34221

2a] function asyncFunction1() {

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

    setTimeout(() => {

      console.log("Async Function 1 completed");

      resolve();

    }, 1000);

  });

}

function asyncFunction2() {

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

    setTimeout(() => {

      console.log("Async Function 2 completed");

      resolve();

    }, 2000);

  });

}

function asyncFunction3() {

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

    setTimeout(() => {

      console.log("Async Function 3 completed");

      resolve();

    }, 3000);

  });

}

asyncFunction1()

  .then(() => {

    return asyncFunction2();

  })

  .then(() => {

    return asyncFunction3();

  })

  .then(() => {

    console.log("All functions completed");

  })

  .catch((error) => {

    console.log(error);

  });

2b]

var number = prompt("Enter a number:");

number = parseInt(number);

if (number % 2 === 0) {

  alert(number + " is an even number.");

} else {

  alert(number + " is an odd number.");

}

3a] function countVowels(str) {

    var vowelCount = 0;

    var lowerCaseStr = str.toLowerCase();

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

      if (lowerCaseStr[i] === 'a' || lowerCaseStr[i] === 'e' || lowerCaseStr[i] === 'i' || lowerCaseStr[i] === 'o' || lowerCaseStr[i] === 'u') {

        vowelCount++;

      }

    }

    return vowelCount;

  }

  var inputString = prompt("Enter a string:");

  var result = countVowels(inputString);

  alert("Number of vowels in the string: " + result);

3b] <!DOCTYPE html>

<html>

<head>

  <title>Sort Letters in Alphabetical Order</title>

</head>

<body>

  <h1>Sort Letters in Alphabetical Order</h1>

  <label for="input">Enter a string:</label>

  <input type="text" id="input">

  <button id="sortButton">Sort</button>

  <script>

    // Function to sort the letters in a string in alphabetical order

    function sortString() {

      // Get the input value

      var input = document.getElementById("input").value;

      // Split the string into an array of characters, sort them, and join them back into a string

      var sortedString = input.split('').sort().join('');

      // Display the sorted string in the console

      alert("Sorted String: " + sortedString);

    }

    // Get the sort button element

    var sortButton = document.getElementById("sortButton");

    // Attach an event listener to the sort button

    sortButton.addEventListener("click", sortString);

  </script>

</body>

</html>

4] <!DOCTYPE html>

<html>

<head>

  <title>Jump Search</title>

</head>

<body>

  <h1>Jump Search</h1>

  <label for="keyInput">Enter the key:</label>

  <input type="number" id="keyInput">

  <label for="numbersInput">Enter a list of numbers (comma-separated):</label>

  <input type="text" id="numbersInput">

  <button id="searchButton">Search</button>

  <script>

    // Function to perform Jump Search on a sorted array

    function jumpSearch(arr, key) {

      var n = arr.length;

      var step = Math.floor(Math.sqrt(n));

      var prev = 0;

      // Find the block where the key might be present

      while (arr[Math.min(step, n) - 1] < key) {

        prev = step;

        step += Math.floor(Math.sqrt(n));

        if (prev >= n) {

          return -1; // Key not found

        }

      }

      // Perform linear search within the block

      while (arr[prev] < key) {

        prev++;

        if (prev === Math.min(step, n)) {

          return -1; // Key not found

        }

      }

      // Check if the key is found

      if (arr[prev] === key) {

        return prev; // Key found at index prev

      }

      return -1; // Key not found

    }

    // Function to handle the search button click event

    function performSearch() {

      // Get the input values

      var key = parseInt(document.getElementById("keyInput").value);

      var numbersInput = document.getElementById("numbersInput").value;

      // Convert the comma-separated string to an array of numbers

      var arr = numbersInput.split(',').map(Number);

      // Perform Jump Search on the array

      var result = jumpSearch(arr, key);

      // Display the search result

      if (result === -1) {

        alert("Key not found.");

      } else {

        alert("Key found at index: " + result);

      }

    }

    // Get the search button element

    var searchButton = document.getElementById("searchButton");

    // Attach an event listener to the search button

    searchButton.addEventListener("click", performSearch);

  </script>

</body>

</html>

5] // Function to encrypt text using Caesar Cipher

function caesarCipherEncrypt(text, shift) {

    var encryptedText = "";

    // Loop through each character in the text

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

      var char = text[i];

      // Check if the character is a letter

      if (char.match(/[a-z]/i)) {

        var code = text.charCodeAt(i);

        // Encrypt uppercase letters

        if (code >= 65 && code <= 90) {

          char = String.fromCharCode(((code - 65 + shift) % 26) + 65);

        }

        // Encrypt lowercase letters

        else if (code >= 97 && code <= 122) {

          char = String.fromCharCode(((code - 97 + shift) % 26) + 97);

        }

      }

      encryptedText += char;

    }

    return encryptedText;

  }

  // Prompt the user to enter the input text

  var inputText = prompt("Enter the text to encrypt:");

  // Prompt the user to enter the shift pattern

  var shiftPattern = parseInt(prompt("Enter the shift pattern (a number between 1 and 25):"));

  // Encrypt the input text using the Caesar Cipher

  var encryptedText = caesarCipherEncrypt(inputText, shiftPattern);

  // Display the encrypted text using an alert

  alert("Encrypted Text: " + encryptedText);

6a] // Function to capitalize the first letter of each word in a string

function capitalizeFirstLetter(str) {

    var words = str.split(' ');

    // Loop through each word in the array

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

      var word = words[i];

      // Capitalize the first letter of the word

      var capitalizedWord = word.charAt(0).toUpperCase() + word.slice(1);

      // Replace the word with the capitalized version in the array

      words[i] = capitalizedWord;

    }

    // Join the words back into a string

    var capitalizedStr = words.join(' ');

    return capitalizedStr;

  }

  // Prompt the user to enter a string

  var inputString = prompt("Enter a string:");

  // Capitalize the first letter of each word in the string

  var capitalizedString = capitalizeFirstLetter(inputString);

  // Display the capitalized string using an alert

  alert("Capitalized String: " + capitalizedString);

6 b] a)onClick()

<!doctype html>

<html>

<head>

    <script>

    function hiThere() {

        alert('Hi there!');

    }

    </script>

</head>

<body>

    <button type="button" onclick="hiThere()">Click me event</button>

</body>

</html>

b)onMouseOver()

<!doctype html>

<html>

<head>

    <script>

    function hov() {

        let e = document.getElementById('hover');

        e.style.display='none'

    }

    </script>

</head>

<body>

    <div id="hover" onmouseover="hov()"

    style="background-color:green;height:200px;width:200px;">

    </div>

</body>

</html>

c)onMouseOut()

<!doctype html>

<html>

<head>

    <script>

    function out() {

        var e = document.getElementById('hover');

        e.style.display = 'none';

    }

    </script>

</head>

<body>

    <div id="hover" onmouseout="out()"

    style="background-color:green;height:200px;width:200px;">

    </div>

</body>

</html>

d)onChange()

<!doctype html>

<html>

<head></head>

<body>

    <input onchange="alert(this.value)" type="number">

</body>

</html>

e)onKeyDown()

<!DOCTYPE html>

<html>

<body>

<input type="text" id="demo" onkeydown="myFunction()">

<script>

function myFunction() {

  document.getElementById("demo").style.backgroundColor = "blue";

}

</script>

</body>

</html>

7] // Function to insert elements into an array using prompt

function insertElements() {

    var arr = [];

    var numElements = parseInt(prompt("Enter the number of elements to insert:"));

    for (var i = 0; i < numElements; i++) {

      var element = parseInt(prompt("Enter element " + (i + 1) + ":"));

      arr.push(element);

    }

    return arr;

  }

  // Function to perform binary search on a sorted array

  function binarySearch(arr, key) {

    var start = 0;

    var end = arr.length - 1;

    while (start <= end) {

      var mid = Math.floor((start + end) / 2);

      if (arr[mid] === key) {

        return mid; // Key found at index mid

      } else if (arr[mid] < key) {

        start = mid + 1;

      } else {

        end = mid - 1;

      }

    }

    return -1; // Key not found

  }

  // Insert elements into the array

  var array = insertElements();

  // Prompt the user to enter the key to search

  var key = parseInt(prompt("Enter the key to search:"));

  // Sort the array in ascending order

  array.sort(function(a, b) {

    return a - b;

  });

  // Perform binary search on the array

  var result = binarySearch(array, key);

  // Display the search result

  if (result === -1) {

    alert("Key not found.");

  } else {

    alert("Key found at index: " + result);

  }

8a] // Function to split a string into an array of words

function splitStringIntoArray(str) {

    // Use the split() method to split the string into an array

    var words = str.split(" ");

    return words;

  }

  // Prompt the user to enter a string

  var inputString = prompt("Enter a string:");

  // Split the string into an array of words

  var wordArray = splitStringIntoArray(inputString);

  // Display the array of words using console.log

  console.log("Array of Words:", wordArray);

8b] <!DOCTYPE html>

<html>

<head>

  <title>VTU USN Validation</title>

</head>

<body>

  <h1>VTU USN Validation</h1>

  <label for="usnInput">Enter VTU USN:</label>

  <input type="text" id="usnInput">

  <button id="validateButton">Validate</button>

  <script>

    // Function to validate VTU USN

    function validateUSN() {

      var usn = document.getElementById("usnInput").value;

      var pattern = /^1[Nv][Ts][0-9]{2}[A-Za-z]{2}[0-9]{3}$/;

      if (pattern.test(usn)) {

        alert("Valid USN");

      } else {

        alert("Invalid USN");

      }

    }

    // Get the validate button element

    var validateButton = document.getElementById("validateButton");

    // Attach an event listener to the validate button

    validateButton.addEventListener("click", validateUSN);

  </script>

</body>

</html>

9a ] // Function to find leap years in a given range

function findLeapYears(startYear, endYear) {

    var leapYears = [];

    for (var year = startYear; year <= endYear; year++) {

      if (year % 4 === 0 && (year % 100 !== 0 || year % 400 === 0)) {

        leapYears.push(year);

      }

    }

    return leapYears;

  }

  // Prompt the user to enter the range of years

  var startYear = parseInt(prompt("Enter the start year:"));

  var endYear = parseInt(prompt("Enter the end year:"));

  // Find the leap years in the given range

  var leapYearsInRange = findLeapYears(startYear, endYear);

  // Display the leap years

  if (leapYearsInRange.length === 0) {

    console.log("No leap years found in the given range.");

  } else {

    console.log("Leap years in the range " + startYear + " to " + endYear + ":");

    console.log(leapYearsInRange);

  }

9b ] // Create a new Set

var mySet = new Set();

// Add values to the Set

mySet.add(10);

mySet.add("Hello");

mySet.add(true);

mySet.add(10); // Adding duplicate value, which will be ignored

// Retrieve value from the Set

var value = "Hello";

if (mySet.has(value)) {

  console.log(value + " is present in the Set.");

} else {

  console.log(value + " is not present in the Set.");

}

// Iterate over the Set

console.log("Values in the Set:");

mySet.forEach(function(item) {

  console.log(item);

});

10 a] // Function to search and display the date within a string

function searchAndDisplayDate(str) {

    var dateRegex = /\d{2}\/\d{2}\/\d{4}/;

    var match = str.match(dateRegex);

    if (match) {

      alert("Date found: " + match[0]);

    } else {

      alert("No date found in the string.");

    }

  }

  // Prompt the user to enter a string

  var inputString = prompt("Enter a string:");

  // Search and display the date within the string

  searchAndDisplayDate(inputString);

10 b] // Create a new Map

var myMap = new Map();

// Add values to the Map

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

myMap.set("age", 30);

myMap.set("city", "London");

// Retrieve value from the Map using key

var key = "age";

if (myMap.has(key)) {

  var value = myMap.get(key);

  console.log("Value for key '" + key + "':", value);

} else {

  console.log("Key '" + key + "' not found in the Map.");

}

// Iterate over the Map

console.log("Entries in the Map:");

myMap.forEach(function(value, key) {

  console.log(key + ":", value);

});

PART B

1) Show how map is different from object to store key value pairs with coding example and prove Maps perform better than objects in most of the scenarios involving addition and removal of keys.

// Using an object

let myObj = {};

// Adding keys

console.time("object-add");

for (let i = 0; i < 100000; i++) {

myObj[i] = "value";

}

console.timeEnd("object-add");

// Removing keys

console.time("object-remove");

for (let i = 0; i < 100000; i++) {

delete myObj[i];

}

console.timeEnd("object-remove");

// Using a map

let myMap = new Map();

// Adding keys

console.time("map-add");

for (let i = 0; i < 100000; i++) {

myMap.set(i, "value");

}

console.timeEnd("map-add");

// Removing keys

console.time("map-remove");

for (let i = 0; i < 100000; i++) {

myMap.delete(i);

}

console.timeEnd("map-remove");

2) Show how set is different from array to store the value with coding example and prove Sets perform better than Arrays in most of the scenarios involving searching values present in it.

// Using an array

let myArray = [];

// Adding values

for (let i = 0; i < 100000; i++) {

myArray.push(i);

}

// Searching for a value

console.time("array-search");

let isValuePresent = myArray.includes(50000);

console.timeEnd("array-search");

// Using a set

let mySet = new Set();

// Adding values

for (let i = 0; i < 100000; i++) {

mySet.add(i);

}

// Searching for a value

console.time("set-search");

isValuePresent = mySet.has(50000);

console.timeEnd("set-search");

3) Implement a JavaScript promise to perform arithmetic operations. Display result for each operation synchronously using await () method. (Give delay in each promise object using settimeout() method).

function add(x, y) {

return new Promise((resolve) => {

setTimeout(() => {

const sum = x + y;

console.log(`Adding ${x} + ${y} = ${sum}`);

resolve(sum);

}, 1000);

});

}

function subtract(x, y) {

return new Promise((resolve) => {

setTimeout(() => {

const diff = x - y;

console.log(`Subtracting ${x} - ${y} = ${diff}`);

resolve(diff);

}, 1500);

});

}

function multiply(x, y) {

return new Promise((resolve) => {

setTimeout(() => {

const product = x \* y;

console.log(`Multiplying ${x} \* ${y} = ${product}`);

resolve(product);

}, 2000);

});

}

function divide(x, y) {

return new Promise((resolve) => {

setTimeout(() => {

const quotient = x / y;

console.log(`Dividing ${x} / ${y} = ${quotient}`);

resolve(quotient);

}, 2500);

});

}

async function performOperations(x, y) {

const sum = await add(x, y);

const diff = await subtract(sum, y);

const product = await multiply(diff, y);

const quotient = await divide(product, y);

return quotient;

}

performOperations(10, 2).then((result) => {

console.log(`Result: ${result}`);

});

4) 4. Develop a Javascript program where user passes the location and a function is called which returns a promise, if the location passed is Paris Below is the output expected: "Let's take a trip to Paris" If the location is other than Paris, show the error message "Invalid Location"

function checkLocation(location) {

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

setTimeout(() => {

if (location.toLowerCase() === 'paris') {

resolve("Let's take a trip to Paris");

} else {

reject('Invalid Location');

}

}, 1000);

});

}

const userLocation = prompt('Enter a location:');

checkLocation(userLocation)

.then((result) => {

console.log(result);

})

.catch((error) => {

console.log(error);

});

5) . Implement a JavaScript program to book a hotel only after booking a flight. [Hint: To achieve this, the promise returned from the bookHotel function is resolved only after resolving the promise from bookFlight function. If the promise gets rejected from bookflight then it won't execute the second function.]

function bookFlight() {

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

setTimeout(() => {

const isFlightBooked = true; // Simulating successful booking

if (isFlightBooked) {

console.log('Flight booked successfully');

resolve();

} else {

reject('Flight booking failed');

}

}, 2000);

});

}

function bookHotel() {

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

setTimeout(() => {

const isHotelBooked = true; // Simulating successful booking

if (isHotelBooked) {

console.log('Hotel booked successfully');

resolve();

} else {

reject('Hotel booking failed');

}

}, 2000);

});

}

bookFlight()

.then(() => {

return bookHotel();

})

.then(() => {

console.log('Everything booked successfully');

})

.catch((error) => {

console.log(error);

});

6)Implement an arrow function that will take one parameter weight in Kg. This arrow function will convert Kg to Lbs. Formula is kg\*2.2

● If LBS is > 150, then the function should return "obese"

● If LBS is between 100 to 150, the function should return "you are ok"

● If LBS is < 100, then the function should return "underweight"

const convertWeight = kg => {

const lbs = kg \* 2.2;

if (lbs > 150) {

return "obese";

} else if (lbs >= 100 && lbs <= 150) {

return "you are ok";

} else {

return "underweight";

}

}

const weightInKg = 75;

const result = convertWeight(weightInKg);

console.log(result); // Output: "you are ok"

7) In the Martian land faraway, a new virus has evolved and is attacking the individuals at a fast pace. The scientists have figured out the virus composition, V. The big task is to identify the people who are infected. The sample of N people is taken to check if they are POSITIVE or NEGATIVE. A report is generated which provides the current blood composition B of the person.

POSITIVE or NEGATIVE?

If the blood composition of the person is a subsequence of the virus composition V, then the person is identified as POSITIVE otherwise NEGATIVE.

Example:

Virus Composition, V = coronavirus

Blood Composition of the person, B = ravus

The person in question is POSITIVE as B is the subsequence of the V.

The scientists are busy with their research for medicine and request you to build a program

which can quickly figure out if the person is POSITIVE or NEGATIVE. They will provide you with

the virus composition V and all the people’s current blood composition. Can you help them?

Note: The virus and blood compositions are lowercase alphabet strings.

function checkInfection(virusComposition, bloodComposition) {

const regex = new RegExp([...bloodComposition].join('.\*'), 'i');

return regex.test(virusComposition) ? 'POSITIVE' : 'NEGATIVE';

}

// Example usage

const virusComposition = 'coronavirus';

const bloodComposition = 'arvus';

const result = checkInfection(virusComposition, bloodComposition);

console.log(result);

8) . Develop a

function isValidEmail(email) {  
var pattern = /^[a-zA-Z0-9!#$%&'\*+\-/=?^\_`{|}~]+(\.[a-zA-Z0-9!#$%&'\*+\-/=?^\_`{|}~]+)\*@[a-zA-Z0-9]+(\.[a-zA-Z0-9]+)\*$/;  
return pattern.test(email);  
}  
  
// Example usage  
var email1 = "mohantg.12#4@gmail.com";  
var email2 = "mohan..tg12#4@gmail.com";  
  
console.log(isValidEmail(email1)); // Output: true  
console.log(isValidEmail(email2)); // Output: fals

9) Implement a JavaScript program using Client-side web APIs to Get the latitude and longitude of the user's position.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>JavaScript Geolocation Demo</title>

</head>

<body>

    <div class="container">

        <button id="show" class="btn">Show my location</button>

        <div id="message"></div>

    </div>

    <script>

        (() => {

            const message = document.querySelector('#message');

            // check if the Geolocation API is supported

            if (!navigator.geolocation) {

                message.textContent = `Your browser doesn't support Geolocation`;

                message.classList.add('error');

                return;

            }

            // handle click event

            const btn = document.querySelector('#show');

            btn.addEventListener('click', function() {

                // get the current position

                navigator.geolocation.getCurrentPosition(onSuccess, onError);

            });

            // handle success case

            function onSuccess(position) {

                const {

                    latitude,

                    longitude

                } = position.coords;

                message.classList.add('success');

                message.textContent = `Your location: (${latitude},${longitude})`;

            }

            // handle error case

            function onError() {

                message.classList.add('error');

                message.textContent = `Failed to get your location!`;

            }

        })();

    </script>

</body>

</html>

10) Develop a JavaScript program to count the number of visitors to keep track of how often a website is accessed and display the number of visitors at the bottom of the homepage.

<!DOCTYPE html>

<html>

<head>

  <title>Visitor Count Example</title>

</head>

<body>

  <h1>Welcome to My Website</h1>

  <p id="visitorCount"></p>

  <script>

    // Check if visitor count exists in localStorage

    let visitorCount = localStorage.getItem('visitorCount');

    if (visitorCount === null) {

      // If visitor count does not exist, initialize it to 1

      visitorCount = 1;

      localStorage.setItem('visitorCount', visitorCount);

    } else {

      // If visitor count exists, increment it by 1

      visitorCount = parseInt(visitorCount) + 1;

      localStorage.setItem('visitorCount', visitorCount);

    }

    // Display the visitor count on the webpage

    document.getElementById('visitorCount').textContent = `Total Visitors: ${visitorCount}`;

  </script>

</body>

</html>