HTML (LAB)

7) a. Write a program using document object properties and methods. .

```
<!DOCTYPE html>
<head>
   <title>Document Object Example</title>
</head>
body>
   <h1 id="heading">Hello, World!</h1>
   <button onclick="changeContent()">Change Heading</button>
   <script>
      function changeContent() {
         // Using document.getElementById to change the content of an element
         document.getElementById("heading").innerHTML = "Content Changed!";
         // Using document.title to change the title of the document
         document.title = "New Page Title";
         // Using document.body.style to change background color
         document.body.style.backgroundColor = "#f0f0f0";
      }
   </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

7) b. Write a program using window object properties and methods

```
<!DOCTYPE html>
<head>>
   <title>Window Object Example</title>
</head>
<body>
   <button onclick="openWindow()">Open Window</button>
   <button onclick="closeWindow()">Close Window</button>
   <script>
      var newWindow;
      function openWindow() {
         // Using window.open() to open a new window
         newWindow = window.open("https://www.example.com", "_blank", "width=600,height=400");
      }
      function closeWindow() {
         // Using window.close() to close the window
         if (newWindow) {
            newWindow.close();
         }   }
   </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

7) c. Write a program using array object properties and methods.

```
<!DOCTYPE html>
<head>
    <title>Array Object Example</title>
    <style>
        body {
            font-family: Arial, sans-serif;
        }
        #output {
            margin-top: 20px;
            padding: 10px;
            border: 1px solid #ccc;
            background-color: #f9f9f9;
        }
    </style>
</head>
<body>
    <h1>Array Operations in JavaScript</h1>
    <div id="output"></div>
    <script>
        // Creating an array
        var numbers = [10, 20, 30, 40, 50];
        // Output div element to show results
        var outputDiv = document.getElementById("output");
        // Using array.length property
        outputDiv.innerHTML += "Array Length: " + numbers.length + "<br><br>";
       // Using array.push() method to add an element to the end of the array
        numbers.push(60);
        outputDiv.innerHTML += "Array after push: " + numbers.join(", ") + "<br>";
        // Using array.pop() method to remove the last element of the array
        var lastElement = numbers.pop();
        outputDiv.innerHTML += "Array after pop: " + numbers.join(", ") + "<br>";
```

HTML (LAB)

```
    outputDiv.innerHTML += "Popped Element: " + lastElement + "<br><br>";
    // Using array.unshift() method to add an element to the beginning of the array
    numbers.unshift(5);
    outputDiv.innerHTML += "Array after unshift: " + numbers.join(", ") + "<br>";
    // Using array.shift() method to remove the first element of the array
    var firstElement = numbers.shift();
    outputDiv.innerHTML += "Array after shift: " + numbers.join(", ") + "<br>";
    outputDiv.innerHTML += "Shifted Element: " + firstElement + "<br><br>";
    // Using array.indexOf() method to find the index of an element
    var index = numbers.indexOf(30);
    outputDiv.innerHTML += "Index of 30: " + index + "<br><br>";
    // Using array.slice() method to extract a part of the array
    var slicedArray = numbers.slice(1, 4);
    outputDiv.innerHTML += "Sliced Array (from index 1 to 4): " + slicedArray.join(", ") + "<br><br>";
    // Using array.join() method to join array elements into a string
    var joinedArray = numbers.join(", ");
    outputDiv.innerHTML += "Array as a string: " + joinedArray + "<br><br>";
    // Using array.reverse() method to reverse the order of the elements in the array
    numbers.reverse();
    outputDiv.innerHTML += "Array after reverse: " + numbers.join(", ") + "<br>";
  </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

7) d. Write a program using math object properties and methods.

```
<!DOCTYPE html>
<head>
  <title>Math Object Example</title>
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    #output {
      margin-top: 20px;
      padding: 10px;
      border: 1px solid #ccc;
      background-color: #f9f9f9;
    }
  </style>
</head>
<body>
  <h1>Math Object Operations in JavaScript</h1>
  <div id="output"></div>
  <script>
    // Output div element to show results
    var outputDiv = document.getElementById("output");
    // Using Math.random() to generate a random number between 0 and 1
    var randomNumber = Math.random();
    outputDiv.innerHTML += "Random Number (between 0 and 1): " + randomNumber +
"<br><br>";
    // Using Math.floor() to round a number down
    var number = 4.9;
    var flooredNumber = Math.floor(number);
    outputDiv.innerHTML += "Math.floor(4.9): " + flooredNumber + "<br><br>";
    // Using Math.ceil() to round a number up
    var ceilNumber = Math.ceil(number);
```

HTML (LAB)

```
    outputDiv.innerHTML += "Math.ceil(4.9): " + ceilNumber + "<br><br>";
    // Using Math.max() to find the maximum value
    var maxValue = Math.max(10, 20, 30, 40, 50);
    outputDiv.innerHTML += "Maximum Value (10, 20, 30, 40, 50): " + maxValue + "<br><br>";
    // Using Math.min() to find the minimum value
    var minValue = Math.min(10, 20, 30, 40, 50);
    outputDiv.innerHTML += "Minimum Value (10, 20, 30, 40, 50): " + minValue + "<br><br>";
    // Using Math.sqrt() to find the square root of a number
    var sqrtValue = Math.sqrt(16);
    outputDiv.innerHTML += "Square Root of 16: " + sqrtValue + "<br><br>";
    // Using Math.pow() to calculate the power of a number
    var powerValue = Math.pow(2, 3); // 2 raised to the power of 3
    outputDiv.innerHTML += "2^3 (2 raised to the power of 3): " + powerValue + "<br><br>";
    // Using Math.round() to round a number to the nearest integer
    var roundedValue = Math.round(4.5);
    outputDiv.innerHTML += "Math.round(4.5): " + roundedValue + "<br><br>";
  </script>
</body>

</html>
```

OUTPUT:

HTML (LAB)

7) e. Write a program using string object properties and methods.

```
<!DOCTYPE html>
<head>
   <title>String Object Example</title>
   <style>
      body {
         font-family: Arial, sans-serif;
      }
      #output {
         margin-top: 20px;
         padding: 10px;
         border: 1px solid #ccc;
         background-color: #f9f9f9;
      }
   </style>
</head>
<body>
   <h1>String Object Operations in JavaScript</h1>
   <div id="output"></div>
   <script>
      // Define a string
      var text = "Hello, JavaScript World!";
      // Output div element to show results
```

HTML (LAB)

```
var outputDiv = document.getElementById("output");

// Using string.length property

outputDiv.innerHTML += "Length of the string: " + text.length + "<br><br>";

// Using string.toUpperCase() method to convert the string to uppercase

outputDiv.innerHTML += "Uppercase: " + text.toUpperCase() + "<br><br>";

// Using string.toLowerCase() method to convert the string to lowercase

outputDiv.innerHTML += "Lowercase: " + text.toLowerCase() + "<br><br>";

// Using string.indexOf() method to find the position of a substring

var index = text.indexOf("JavaScript");

outputDiv.innerHTML += "Index of 'JavaScript': " + index + "<br><br>";

// Using string.slice() method to extract a part of the string

var slicedString = text.slice(7, 18); // Extracts substring from index 7 to 18

outputDiv.innerHTML += "Sliced String (7 to 18): " + slicedString + "<br><br>";

// Using string.replace() method to replace a part of the string

var replacedString = text.replace("World", "Universe");

outputDiv.innerHTML += "Replaced String: " + replacedString + "<br><br>";

// Using string.split() method to split the string into an array

var splitArray = text.split(" ");

outputDiv.innerHTML += "String split into an array: [" + splitArray.join(", ") + "]<br><br>";

// Using string.charAt() method to get the character at a specific index

var charAtPosition = text.charAt(6);

outputDiv.innerHTML += "Character at index 6: " + charAtPosition + "<br><br>";

// Using string.trim() method to remove whitespace from both ends

var stringWithSpaces = "   Trim this string!   ";

var trimmedString = stringWithSpaces.trim();
```

HTML (LAB)

```
        outputDiv.innerHTML += "Trimmed String: '" + trimmedString + "'<br><br>";

        // Using string.includes() method to check if a substring exists
        var containsWord = text.includes("JavaScript");
        outputDiv.innerHTML += "Does the string include 'JavaScript'? " + containsWord + "<br><br>";
    </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

7) f. Write a program using regex object properties and methods.

```
<!DOCTYPE html>
<html lang="en">
  <title>RegEx Object Example</title>
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    #output {
      margin-top: 20px;
      padding: 10px;
      border: 1px solid #ccc;
      background-color: #f9f9f9;
    }
  </style>
</head>
<body>
  <h1>RegEx Object Operations in JavaScript</h1>
  <div id="output"></div>
  <script>
    // Define a string to work with
    var text = "The quick brown fox jumps over the lazy dog 1234567890";
    // Output div element to show results
    var outputDiv = document.getElementById("output");
    // Create a regular expression using RegExp constructor
    var regex = new RegExp("fox");
    // Using regex.test() method to test if a string matches the pattern
    var testResult = regex.test(text);
    outputDiv.innerHTML += "Test for 'fox': " + testResult + "<br><br>";
    // Using regex.exec() method to find the first match of the pattern in the string
    var execResult = regex.exec(text);
    outputDiv.innerHTML += "Exec result for 'fox': " + (execResult ? execResult[0] : "No match") +
"<br><br>";
    // Create a regex pattern to find all digits
    var digitRegex = /¥d+/g;
```

HTML (LAB)

```
    // Using string.match() method to find all matches of a regular expression in a string
    var matchResult = text.match(digitRegex);
    outputDiv.innerHTML += "Match digits in text: " + matchResult.join(", ") + "<br><br>";
    // Using string.replace() method with a regular expression to replace all vowels with an asterisk
    var replaceResult = text.replace(/[aeiouAEIOU]/g, "*");
    outputDiv.innerHTML += "Replace vowels with '*': " + replaceResult + "<br><br>";
    // Using string.search() method to find the index of the first match of a regular expression
    var searchResult = text.search(/[A-Za-z]+/);
    outputDiv.innerHTML += "Search for first word (letters only): " + searchResult + "<br><br>";
    // Using regex.test() to check if the string contains any uppercase letters
    var uppercaseRegex = /[A-Z]/;
    var hasUppercase = uppercaseRegex.test(text);
    outputDiv.innerHTML += "Contains uppercase letters: " + hasUppercase + "<br><br>";
    // Using string.split() with a regex to split the string based on spaces
    var splitResult = text.split(/¥s+/);
    outputDiv.innerHTML += "Split string by spaces: [" + splitResult.join(", ") + "]<br><br>";
  </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

7)g. Write a program using date object properties and methods.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Date Object Program</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      padding: 20px;
    }
    .output {
      margin-bottom: 10px;
    }
  </style>
</head>
<body>
  <h1>Date Object Properties and Methods</h1>
  <div class="output" id="currentDateTime"></div>
  <div class="output" id="year"></div>
  <div class="output" id="month"></div>
  <div class="output" id="dayOfMonth"></div>
  <div class="output" id="dayOfWeek"></div>
  <div class="output" id="hours"></div>
  <div class="output" id="minutes"></div>
  <div class="output" id="seconds"></div>
  <div class="output" id="milliseconds"></div>
  <div class="output" id="formattedDate"></div>
  <script>
    // Create a new Date object
    const currentDate = new Date();
    // Display the current date and time
    document.getElementById("currentDateTime").textContent = "Current Date and Time: " + currentDate;
    // Get the full year
    document.getElementById("year").textContent = "Year: " + currentDate.getFullYear();
```

HTML (LAB)

```
    // Get the month (0-based index)
    document.getElementById("month").textContent = "Month: " + (currentDate.getMonth() + 1); // Adding 1 to
make it 1-based
    // Get the day of the month
    document.getElementById("dayOfMonth").textContent = "Day of the month: " + currentDate.getDate();
    // Get the day of the week (0 - Sunday, 6 - Saturday)
    document.getElementById("dayOfWeek").textContent = "Day of the week: " + currentDate.getDay();
    // Get the hours
    document.getElementById("hours").textContent = "Hours: " + currentDate.getHours();
    // Get the minutes
    document.getElementById("minutes").textContent = "Minutes: " + currentDate.getMinutes();
    // Get the seconds
    document.getElementById("seconds").textContent = "Seconds: " + currentDate.getSeconds();
    // Get the milliseconds
    document.getElementById("milliseconds").textContent = "Milliseconds: " + currentDate.getMilliseconds();
    // Format the date as a string (e.g., "Thu Dec 12 2024 12:34:56 GMT+0000")
    document.getElementById("formattedDate").textContent = "Formatted Date: " + currentDate.toString();
  </script>
</body>
</html>
```

OUTPUT:

HTML (LAB)

8) h. Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.

```
< !DOCTYPE html>
  <html lang="en">
  <head>
    <title>User-Defined Object Example</title>
  </head>
  <body>
    <h1>User-Defined Object in JavaScript</h1>
    <p>Google Chrome/Edge: Press Ctrl+Shift+I (Windows/Linux) or Cmd+Option+I (Mac), then go to the "Console" tab.
      Open the console to see the output of the script demonstrating user-defined objects in JavaScript.</p>
    <script>
      // Define a user-defined object using a class
      class Car {
        // Constructor
        constructor(make, model, year) {
          this._make = make; // Private-like property
          this._model = model; // Private-like property
          this._year = year; // Private-like property
        }
        // Getter for make
        get make() {
          return this._make;
        }
        // Setter for make
        set make(value) {
          if (!value) {
            throw new Error("Make cannot be empty");
          }
          this._make = value;
        }
        // Getter for model
        get model() {
          return this._model;
        }
        // Setter for model
        set model(value) {
          if (!value) {
            throw new Error("Model cannot be empty");
          }
          this._model = value;
        }
        // Getter for year
        get year() {
          return this._year;
        }
        // Setter for year
        set year(value) {
          if (value < 1900 || value > 2100) {
            throw new Error("Year must be between 1900 and 2100");
          }
```

HTML (LAB)

```
        this._year = value;
      }

      // Method to display car details
      display() {
        console.log(`Car Details: ${this._year} ${this._make} ${this._model}`);
      }

      // Method to check if the car is vintage
      isVintage() {
        return this._year < 1990;
      }
    }
    // Create an instance of the Car class
    const car1 = new Car("Toyota", "Corolla", 2020);
    // Access and modify properties
    console.log("Initial Details:");
    car1.display();
    console.log("\nModifying Details:");
    car1.make = "Honda";
    car1.model = "Civic";
    car1.year = 2018;
    car1.display();
    // Use a method
    if (car1.isVintage()) {
      console.log("This car is vintage.");
    } else {
      console.log("This car is not vintage.");
    }
  </script>
</body>
</html>
```

OUTPUT: