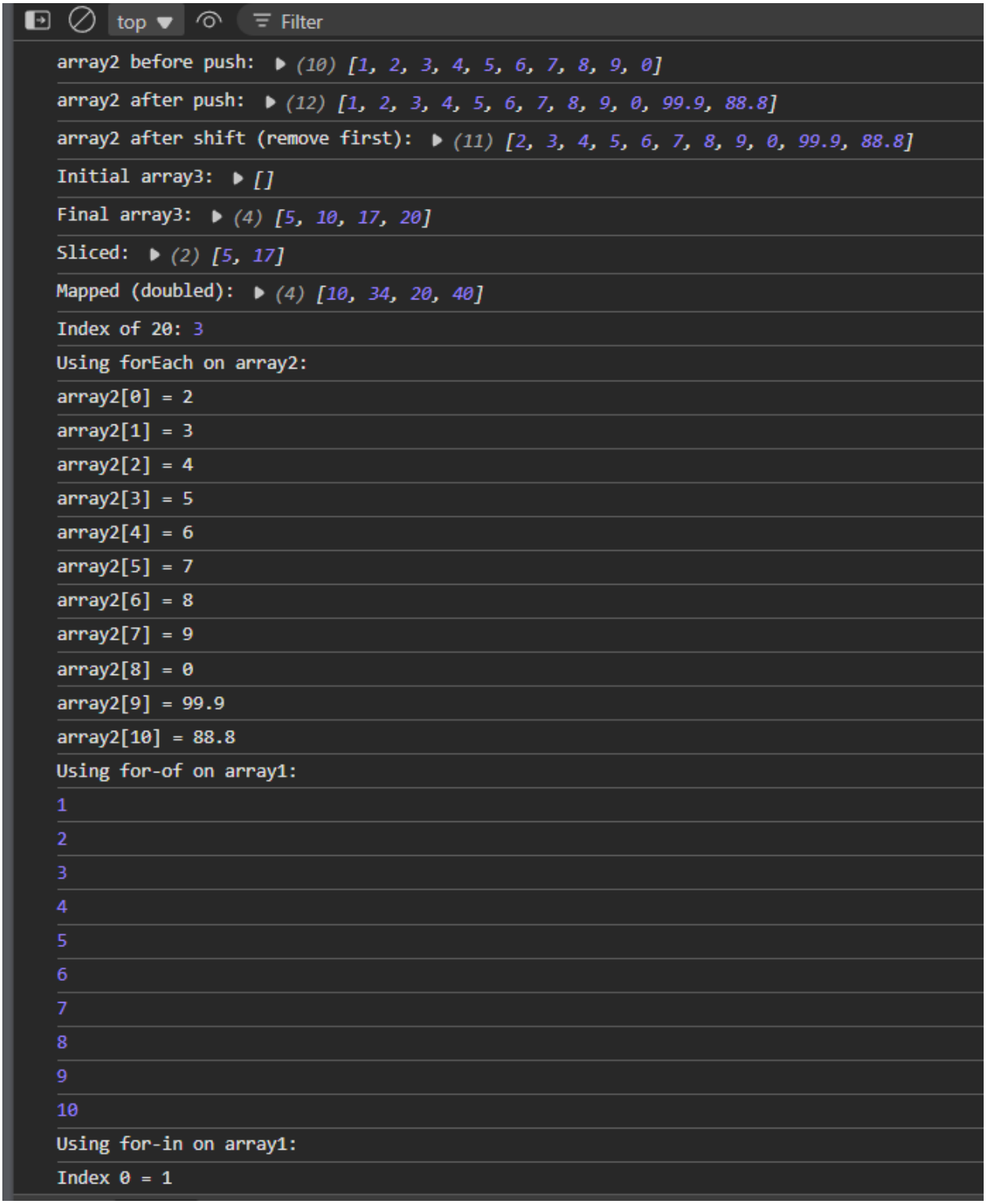
22.a

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Array and JavaScript Tasks</title>
</head>
<body>
   <h2>Open your browser console to view output</h2>

   <script>
      // 1. array1: 10 natural numbers using initialization
      let array1 = [1,2,3,4,5,6,7,8,9,10];

      // 2. array2: 10 float numbers taken via prompt
      let array2 = [];
      for(let i=0; i<10; i++) {
         let num = parseFloat(prompt(`Enter floating point number ${i+1}:`));
         array2.push(num);
      }

      // 3. array3: empty array
      let array3 = [];

      // Sum of elements in array1 and check if divisible by 7
      let sum1 = 0;
      for(let i=0; i<array1.length; i++) {
         sum1 += array1[i];
      }
      if(sum1 % 7 === 0){
         alert(`Sum of array1 is ${sum1}, and it IS divisible by 7`);
      } else {
         alert(`Sum of array1 is ${sum1}, and it is NOT divisible by 7`);
      }
```

```
    // Add two elements at the end of array2
    console.log("array2 before push:", [...array2]);       // This creates a new
array with the current values at the time of logging.

    array2.push(99.9, 88.8);
    console.log("array2 after push:", [...array2]);

    // Delete first element
    array2.shift();
    console.log("array2 after shift (remove first):", [...array2]);

    // Sum of digits of number (e.g., 1457 => 1+4+5+7 = 17)
    let numInput = prompt("Enter a number to calculate sum of digits:");
    let sumDigits = 0;
    let temp = parseInt(numInput);
    while(temp !== 0) {
       sumDigits += temp % 10;
       temp = Math.floor(temp / 10);
    }
    alert(` Sum of digits = ${sumDigits}` );

    // Take numbers using prompt + confirm until user stops, then sum them
    let totalSum = 0;
    let input;
    do {
       input = parseFloat(prompt("Enter a number:"));
       totalSum += input;
    } while (confirm("Do you want to enter another number?"));
    alert(` Total sum of entered numbers: ${totalSum}` );

    // Use 10 array methods on array3
    console.log("Initial array3:", array3);
    array3.push(10);                // 1. push
    array3.push(20, 30);              // 2. push
```

```
        array3.unshift(5);              // 3. unshift
        array3.pop();                   // 4. pop
        array3.splice(1, 0, 15);        // 5. splice
        array3[1] = 17;                 // 6. modify directly
        let sliced = array3.slice(0, 2);   // 7. slice
        let mapped = array3.map(x => x * 2); // 8. map
        let index = array3.indexOf(20);   // 9. indexOf
        array3.sort((a,b)=>a-b);        // 10. sort
        console.log("Final array3:", array3);
        console.log("Sliced:", sliced);
        console.log("Mapped (doubled):", mapped);
        console.log("Index of 20:", index);

        // Use forEach on array2
        console.log("Using forEach on array2:");
        array2.forEach((value, index) => {
            console.log(`array2[${index}] = ${value}`);
        });

        // Use for-of and for-in on array1
        console.log("Using for-of on array1:");
        for(let val of array1){
            console.log(val);
        }

        console.log("Using for-in on array1:");
        for(let index in array1){
            console.log(`Index ${index} = ${array1[index]}`);
        }
    </script>
</body>
</html>
```


top
Filter
array2 before push: ▶ (10) [1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
array2 after push: ▶ (12) [1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 99.9, 88.8]
array2 after shift (remove first): ▶ (11) [2, 3, 4, 5, 6, 7, 8, 9, 0, 99.9, 88.8]
Initial array3: ▶ []
Final array3: ▶ (4) [5, 10, 17, 20]
Sliced: ▶ (2) [5, 17]
Mapped (doubled): ▶ (4) [10, 34, 20, 40]
Index of 20: 3
Using forEach on array2:
array2[0] = 2
array2[1] = 3
array2[2] = 4
array2[3] = 5
array2[4] = 6
array2[5] = 7
array2[6] = 8
array2[7] = 9
array2[8] = 0
array2[9] = 99.9
array2[10] = 88.8
Using for-of on array1:
1
2
3
4
5
6
7
8
9
10
Using for-in on array1:
Index 0 = 1

22.b

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        function func1(){
            document.write("I am invoked <b>normal definition</b><br>");
        }
        const func2= function(){
            document.write('I am invoked <b>function expression</b><br>')
        }
        const func3= ()=> {
            document.write('I am invoked <b>arrow function</b><br>')
        }
        const func4= (a,b) => {
            return a+b;
        }
        const func5= (a,b) => a%b; // single line arrow function

        func1(); // normal function invocation
        func2(); // function expression invocation
        func3(); // arrow function invocation
        document.write(` Sum of 2 and 3 is: ${func4(2,3)}<br>` ); // normal function
invocation
        document.write(` Modulus of 31 and 11 is: ${func5(31,11)}<br>` ); // single line
arrow function invocation
    </script>
</head>
<body>

</body>
</html>
```

I am invoked **normal definition**
I am invoked **function expression**
I am invoked **arrow function**
Sum of 2 and 3 is: 5
Modulus of 31 and 11 is: 9

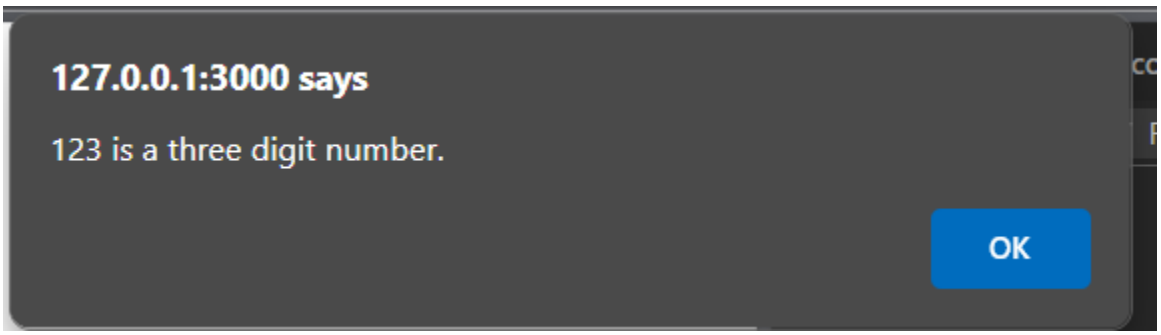
22.c

```
<script>
    // creaing object using object literal syntax
    player= {
        name: 'messi',
        age: 38,
        country: 'argentina',
        club: 'inter miami'
    }
    // add new property to object
    player.foot = 'left';
    player.show= function(){
        console.log(this.name + ' ' + this.age + ' ' + this.country + ' ' + this.club + ' '
+ this.foot);
    }
    player.show();

    // creating instance of object using new keywored
    house = new Object();
    house.color = 'red';
    house.size = 'big';
    house.location = 'newyork';
    house['rooms'] = 5;
    house.show = function(){
        console.log(this.color + ' ' + this.size + ' ' + this.location + ' ' + this.rooms);
    }
    house.show();

    // deleting property from object
```

```
    delete house.location
    console.log(house.location); // undefined

    // creating object using constructor function
    function Car(model, year, fuel){
        this.model = model;
        this.year = year;
        this.fuel = fuel;
        this.show = function(){
            console.log(this.model + ' ' + this.year + ' ' + this.fuel);
        }
    }
    // creating many instances of object using constructor function
    toyata = new Car('toyota', 2020, 'petrol');
    honda = new Car('honda', 2021, 'diesel');
    tesla= new Car('tesla', 2022, 'electric');
    toyata.show();
</script>
```

22.d

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        let num= Math.abs(parseInt((prompt("Enter any integer: "))));
        let digitCount=0;
        const n= num;
        while (num > 0){
            num = Math.floor(num/10);
            digitCount++;
            document.write(digitCount + "<br>");
        }
        if (digitCount == 1 | digitCount == 0){
            alert(` ${n} is a single digit number.` );
        } else if (digitCount == 2){
            alert(` ${n} is a double digit number.` );
        } else if (digitCount == 3){
            alert(` ${n} is a three digit number.` );
        } else {
            alert(` ${n} is a multi digit number.` );
        }
    </script>
</head>
<body>
</body>
```



22.e

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        let num1, num2;
        do{
            num1= parseFloat(prompt("Enter first number: "));
            num2= parseFloat(prompt("Enter second number: "));
            operator= prompt("Enter operator (+, -, *, /): ");

            switch(operator){
                case '+':
                    result= num1 + num2;
                    alert(num1 + " + " + num2 + " = " + result);
                    break;
                case '-':
                    result= num1 - num2;
                    alert(num1 + " - " + num2 + " = " + result);
                    break;
                case '*':
                    result= num1 * num2;
                    alert(num1 + " * " + num2 + " = " + result);
                    break;
                case '/':
                    if(num2 !== 0){
                        result= num1 / num2;
                        alert(num1 + " / " + num2 + " = " + result);
                    } else {
                        alert("Error: Division By Zero.");
                    }
```

```
                break;
            default:
                alert("Invalid operator.");
        }
    }
    while(confirm("Do you want to perform another calculation?"));
  </script>

</head>
<body>

</body>
</html>
```

22.f

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        let divident = 45, divisor = 0;
        try {
            if (divisor == 0){
                throw new Error("Division by zero is not allowed.");
            }
            let result= divident / divisor;
            document.write("Result: " + result + "<br>");
        }
        catch (error) {
            document.write("<br>Error: " + error.message + "<br>");
        }
        finally {
            document.write("Finally block executed <br>");
        }
        document.write("End of script execution");
    </script>
</head>
<body>

</body>
</html>
```

```
Error: Division by zero is not allowed.
Finally block executed
End of script execution
```

23.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        document.write(`Random Integer: ${Math.round(Math.random()*10)}`);
        const str1= "Hello";
        const str2= "World";
        document.write("<br>Concatenated String(lwr, upr): " +
str1.toLowerCase().concat(' ' + str2.toUpperCase()));

    </script>
</head>
<body>

</body>
</html>
```

```
Random Integer: 9
Concatenated String(lwr, upr): hello WORLD
```

24.a

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script>
        console.log(Date())
        console.log(new Date())
        console.log(new Date('january 22, 2012, 2:30:44'))
        console.log(new Date(2010,12,2,33,11,45))    //months are 0-based in the Date(year, monthIndex, day) constructor. 0 = January, 12 = January of the next year.
        // automatically handles overflow date excedding max limit reuslt in next month
        console.log(new Date(0))
        console.log(new Date(24*60*60*1000)) // in miliseconds
    </script>
</head>
<body>

</body>
</html>
```

24.b

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Document</title>
</head>
<body>
   <script>
      d= new Date()
      document.write(` Today's date: ${d}<br>
      <br>Getter Methods: <br>
      milli-seconds: ${d.getTime()}<br>
      Day of month: ${d.getDate()}<br>
      Day of week: ${d.getDay()}<br>
      Full Year: ${d.getFullYear()}<br>
      JSON: ${d.toJSON()}<br>` )

      document.write(`<br>Setter Methods: <br>` )
      d.setFullYear(2030)
      d.setDate(2)
      d.setMonth(11)
      document.write(` Date: ${d}<br>` )
      document.write('<br>Note: some getter setter methods works on
index<br>')
   </script>
</body>
</html>
```

```
Today's date: Thu Apr 10 2025 22:16:26 GMT+0545 (Nepal Time)

Getter Methods:
milli-seconds: 1744302686154
Day of month: 10
Day of week: 4
Full Year: 2025
JSON: 2025-04-10T16:31:26.154Z

Setter Methods:
Date: Mon Dec 02 2030 22:16:26 GMT+0545 (Nepal Time)

Note: some getter setter methods works on index
```

24.c

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Calculate your age</h1>
    <label for="birth_date">Enter your birth date:</label>
    <input type="date" id="birth_date" value="2017-10-24" min="1900-01-
01"/><br>
    <button onclick="getDate()">Submit</button>
    <p id="years"> </p>
    <p id="full-age"> </p>
    <br>
    <script>
        document.getElementById("birth_date").max= new
Date().toISOString().split('T')[0]; // Set max date to today
        function getDate(){
            const birth_date= document.getElementById('birth_date').value;
            const b_date_obj= new Date(birth_date);
            const today= new Date();
            let age= today.getFullYear() - b_date_obj.getFullYear();
            let month= today.getMonth() - b_date_obj.getMonth();
            let day= today.getDate() - b_date_obj.getDate();

            if (month < 0 || (month ===0 && day < 0)){   // few months shorter than a
year
                age--;
                month+= 12;
            }
            if (day < 0){   // few days shorter than a month
                month --;
                const lastMonth = new Date(today.getFullYear(), today.getMonth(), 0);
```

```
            day+= lastMonth.getDate();
        }

        document.getElementById("years").innerHTML= `You're <b>${age}</b>
years old.`;
        document.getElementById("full-age").innerHTML= `You're <b>${age}</b>
years, <b>${month}</b> months and <b>${day}</b> days old.`;
    }

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

Calculate your age

Enter your birth date: 01/21/2009

Submit

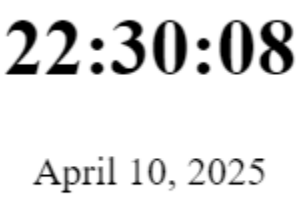
You're **16** years old.

You're **16** years, **2** months and **20** days old.

24.d

```
<!-- Create a web page having a digital clock blinking each second and displaying correct time. The page also should display the current date.  -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <style>
    body{
      text-align: center;
    }

    .blink {
      visibility: hidden;
    }
  </style>
</head>

<body>
  <div id="time"> </div>
  <p id="date"> </p>
  <script>
    function updateClock() {
      const now = new Date();
      const hours = String(now.getHours()).padStart(2, '0');
      const minutes = String(now.getMinutes()).padStart(2, '0');
      const seconds = String(now.getSeconds()).padStart(2, '0');
      const date = now.toLocaleDateString('nep', { year: 'numeric', month: 'long', day: 'numeric' });

      /*
      // Format the time (HH:MM:SS)
```

```
        hours = (hours < 10 ? '0' : '') + hours;
        minutes = (minutes < 10 ? '0' : '') + minutes;
        seconds = (seconds < 10 ? '0' : '') + seconds; */

        // Display the current time
        document.getElementById('time').innerHTML =
`<h1>${hours}:${minutes}:${seconds}</h1>`;
        console.log(` ${hours}:${minutes}:${seconds} ${date}` );
        if (seconds % 2 === 0 )
           document.getElementById('time').classList.remove("blink");
        else
           document.getElementById('time').classList.add("blink");

        // display the date
        document.getElementById('date').textContent = `${date}`;
      }

      setInterval(updateClock, 1000);
      updateClock(); // Initial call to display immediately
   </script>
 </body>
 </html>
```



April 10, 2025

24.e

```
Given Date and Time: Thu Apr 10 2025 14:30:00 GMT+0545 (Nepal Time)
Before (2 days, 5 hours): Tue Apr 08 2025 09:30:00 GMT+0545 (Nepal Time)
After (2 days, 5 hours): Sat Apr 12 2025 19:30:00 GMT+0545 (Nepal Time)
```

```
<!DOCTYPE htm
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <script>
    function displayBeforeAfter(dateStr, days, hours) {
      const givenDate = new Date(dateStr);

      // Create a copy of the date to manipulate
      const msInHour = 60 * 60 * 1000;
      const msInDay = 24 * msInHour;
      const offsetMs = (days * msInDay) + (hours * msInHour);

      const beforeDate = new Date(givenDate.getTime() - offsetMs);
      const afterDate = new Date(givenDate.getTime() + offsetMs);

      console.log("Given Date and Time:", givenDate.toString());
      console.log(` Before (${days} days, ${hours} hours):` ,
beforeDate.toString());
      console.log(` After (${days} days, ${hours} hours):` , afterDate.toString());
    }

    // Example usage:
    const inputDate = "2025-04-10T14:30"; // ISO format works best
    const days = 2;
    const hours = 5;
    displayBeforeAfter(inputDate, days, hours);
  </script>
</body>
```

24.f

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Stopwatch</title>
    <style>
        #container {
            text-align: center;
            font-family: Arial, sans-serif;
            border: 2px solid #000;
            width: 50%;
            justify-self: center;
        }

        #watch {
          font-size: 2em;
          margin-bottom: 10px;
        }
        button {
          margin-right: 5px;
          padding: 8px 12px;
        }
        #lapsContainer {
            margin: 20px;
            text-align: center;
            /* border: 2px solid #000; */

        }
        ol{
            list-style-position: inside;
            padding: 0;
            margin: 20px;
```

```
      }
    </style>
</head>
<body>
   <div id ="container">
      <h1>Stopwatch</h1>
      <div id="watch"> </div>
      <div id="buttons">
         <button id="toggle" onclick="toggleWatch()">Start</button>
         <!-- <button id="stop" onclick="stopWatch()">Stop</button> -->
         <button id="reset" onclick="resetWatch()">Reset</button>
         <button id="lap" onclick="lapTime()">Lap</button>
      </div>
      <div id="lapsContainer">
         <h3>Laps</h3>
         <ol id="laps"> </ol>
      </div>
   </div>
   <script>
      let hours=0, minutes=0, seconds = 0;
      let interval = null; // Variable to store the interval ID
      document.getElementById('watch').innerHTML = `00:00:00`;
      function updateWatch() {
         seconds ++; // increment seconds by 1
         if (seconds === 60) {
            seconds = 0; // reset seconds to 0
            minutes ++; // increment minutes by 1

            if (minutes === 60) {
               minutes = 0; // reset minutes to 0
               hours ++; // increment hours by 1

               if (hours === 24) {
                  hours = 0; // reset hours to 0
               }
```

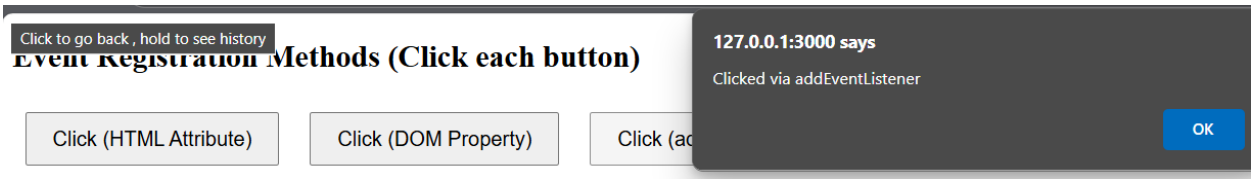
```
            }
        }

        const formattedTime=
            hours.toString().padStart(2, '0') + ':' +
            minutes.toString().padStart(2, '0') + ':' +
            seconds.toString().padStart(2, '0'); // format the time as HH:MM:SS

        document.getElementById('watch').innerHTML = formattedTime; // update
the watch display`;
    }

    function toggleWatch() {
        toggleButton= document.getElementById('toggle'); // Get the start button
element

        if (!interval) { // Start the stopwatch only if it's not already running.
            interval = setInterval(updateWatch, 1000); // call updateWatch every
second
            toggleButton.textContent = 'Pause'; // Change the button text to
"Pause"
        }
        else {
            stopWatch(); // Call the stopWatch function to stop the stopwatch
            toggleButton.textContent = 'Resume'; // Change the button text back to
"Start"
        }
    }

    function stopWatch() {
        clearInterval(interval); // Stop the stopwatch
        interval = null; // Reset the interval variable
    }

    function resetWatch() {
        stopWatch(); // Stop the stopwatch
```

```
        hours= minutes = seconds = 0; // Reset hours, minutes, and seconds to 0
        document.getElementById('toggle').textContent = 'Start'; // Change the
button text back to "Start"
        document.getElementById('watch').textContent= '00:00:00'; // Update the
display to show 00:00:00
        document.getElementById('laps').textContent = ``;
    }

    function lapTime() {
        const lapTime= document.getElementById('watch').textContent; // Get the
current time from the stopwatch display
        const li= document.createElement('li'); // Create a new list item for the lap
time
        li.textContent= lapTime; // Set the text content of the list item to the
current time
        document.getElementById('laps').appendChild(li); // Append the new list item
to the laps list

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

25.a

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Event Registration in JavaScript</title>
  <style>
    button {
      margin: 10px;
      padding: 10px 20px;
      font-size: 16px;
    }
  </style>
</head>
<body>

  <h2>Event Registration Methods (Click each button)</h2>

  <!-- 1. Inline HTML Event Attribute -->
  <button onclick="alert('Clicked via HTML attribute')">Click (HTML
Attribute)</button>

  <!-- 2. DOM Element Property (set via JS) -->
  <button id="btnProperty">Click (DOM Property)</button>

  <!-- 3. addEventListener Method -->
  <button id="btnListener">Click (addEventListener)</button>

  <script>
    // 2. Registering with DOM element property
    document.getElementById('btnProperty').onclick = function () {
      alert('Clicked via DOM property');
    };

    // 3. Registering with addEventListener
```

```
   document.getElementById('btnListener').addEventListener('click', function ()
{
     alert('Clicked via addEventListener');
   });
  </script>

</body>
</html>
```

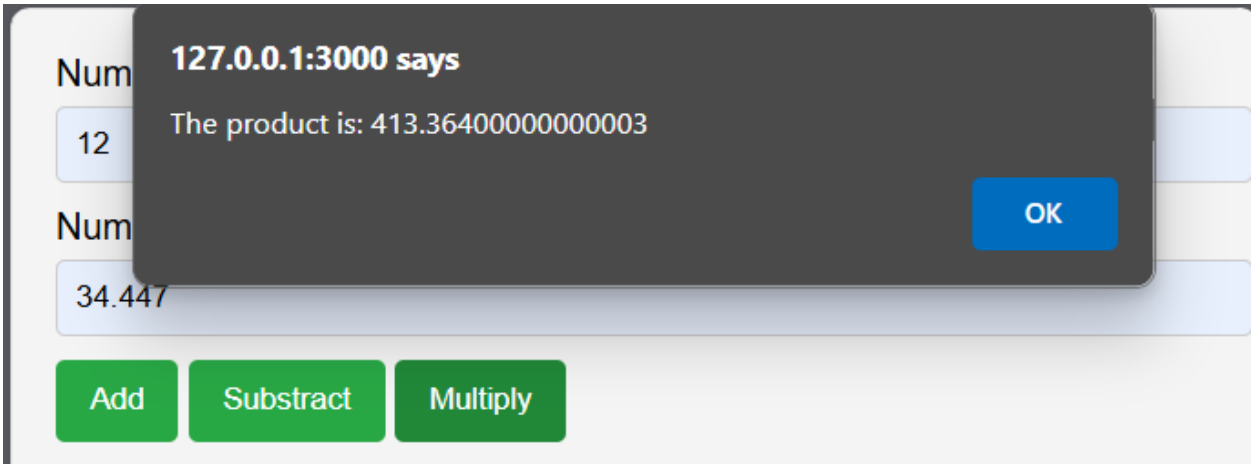


25.b

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 20px;
        }

        label {
            display: block;
            margin-bottom: 5px;
        }

        input[type="numeric"] {
            width: 100%;
            padding: 8px;
            margin-bottom: 10px;
            border: 1px solid #ccc;
            border-radius: 4px;
        }

        button {
            padding: 10px 15px;
            background-color: #28a745;
            color: white;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }
```

```
        button:hover {
            background-color: #218838;
        }
    </style>

</head>
<body>
    <label for="num1">Number 1</label>
    <input type="numeric" id="num1" placeholder="Enter any number"
required><br>
    <label for="num2">Number 2</label>
    <input type="numeric" id="num2" placeholder="Enter any number"
required><br>

    <button id="add" onclick="Add()">Add</button>
    <button id="substract" onclick="Substract()">Substract</button>
    <button id="multiply" onclick="Multiply()">Multiply</button>

    <script>
        function getValues() {
            num1= parseFloat(document.getElementById("num1").value);
            num2= parseFloat(document.getElementById("num2").value);

            if (isNaN(num1) || isNaN(num2)) {
                alert("Please enter valid numbers in both fields.");
                return;
            }
        }

        function Add(){
            getValues(); // Call the function to get the values before performing the
operation
            alert("The sum is: " + (num1 + num2));
        }
```

```
        function Substract() {
            getValues(); // Call the function to get the values before performing the
operation
            alert("The difference is: " + (num1 - num2));
        }

        function Multiply() {
            getValues(); // Call the function to get the values before performing the
operation
            alert("The product is: " + (num1 * num2));
        }
        // Add event listeners to the buttons
    </script>

</body>
</html>
```



25.c-g

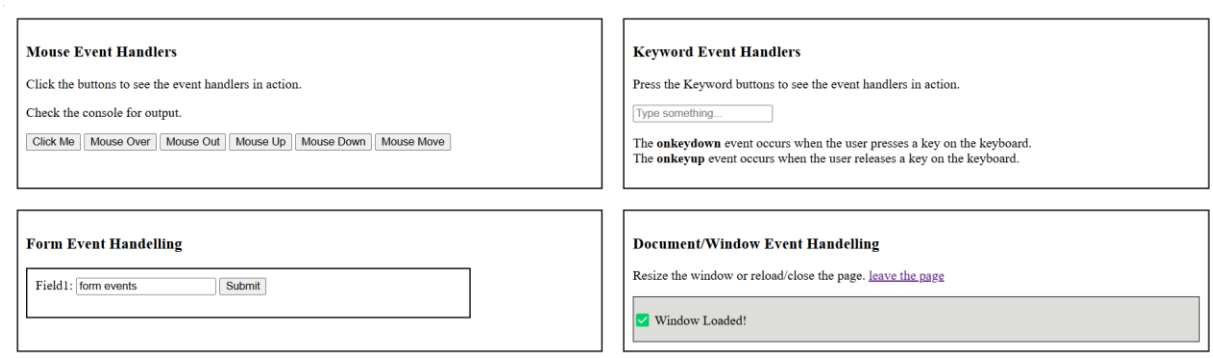
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        body{
            display: flex;
            flex-wrap: wrap;
            gap: 5px;
        }
        div.event {
            border: 2px solid black;
            width: 45vw;
            margin: 10px;
            padding: 10px;
        }

        form{
            padding: 10px;
            border: 2px solid;
            width: 75%;
        }
        #log{
            border: 1px solid;
            background-color: rgb(221, 221, 218);
            max-height: 200px;
            overflow-y: auto;
            width: auto;
        }
    </style>
</head>
<body>
```

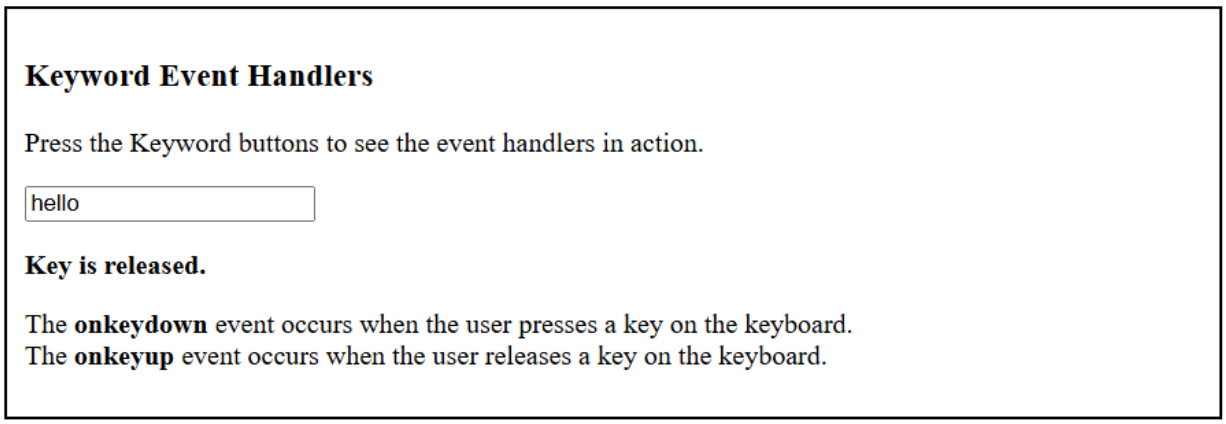
```
  <div class="event">
    <h3>Mouse Event Handlers</h3>
    <p>Click the buttons to see the event handlers in action.</p>
    <p>Check the console for output.</p>
    <button id="click" onclick="console.log('Clicked!')">Click Me</button>
    <button id="over" onmouseover="console.log('Mouse Over')">Mouse
Over</button>
    <button id="out" onmouseout="console.log('mouse out')">Mouse
Out</button>
    <button id="up" onmouseup="console.log('Mouse Up')">Mouse Up</button>
    <button id="down" onmousedown="console.log('mouse down')">Mouse
Down</button>
    <button id="move" onmousemove="console.log('mouse move')">Mouse
Move</button>
  </div>

  <div class='event'>
    <h3>Keyword Event Handlers</h3>
    <p>Press the Keyword buttons to see the event handlers in
action.</p>
    <input type="text" id="keyword" placeholder="Type something..."
    onkeydown="document.getElementById('note').innerHTML='<b>Key is
pressed.</b>'; "
    onkeyup="document.getElementById('note').innerHTML='<b>Key is
released.</b>'; "
    onfocus="this.placeholder='Welcome Back!!!'"/>
    <p id="note"></p>

    <p>The <b>onkeydown</b> event occurs when the user presses a key on the
keyboard. <br>
    The <b>onkeyup</b> event occurs when the user releases a key on the
keyboard.</p>
  </div>

  <div class="event">
```

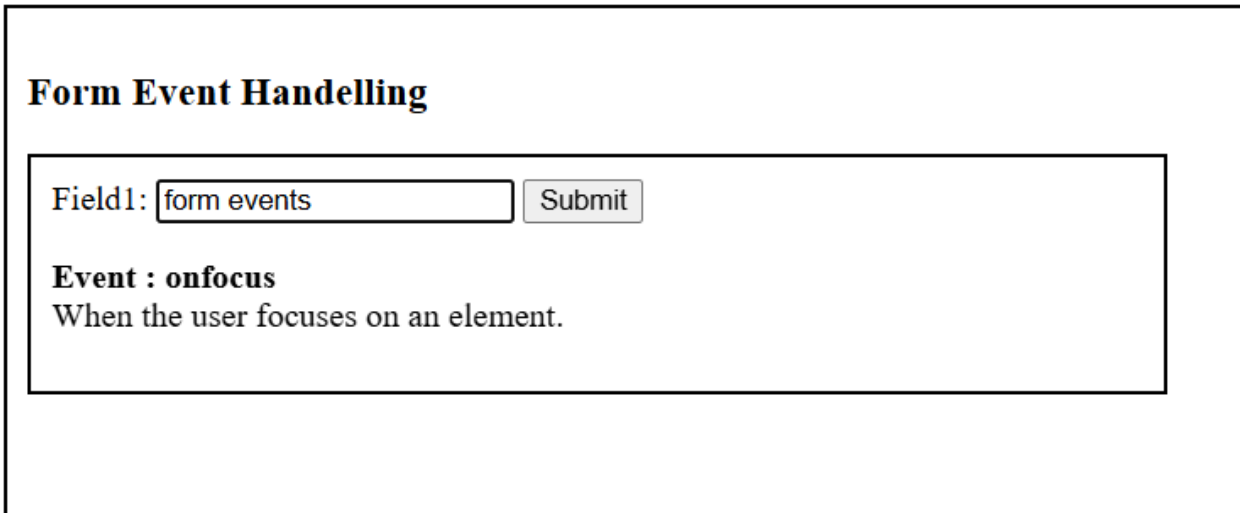
```
    <h3>Form Event Handelling</h3>
    <form onsubmit="FormEvent('submit'); return false">
      Field1:
      <input type="text" onfocus="FormEvent('focus')" onblur="handleBlur()"
oninput="FormEvent('input')" onchange="handleChange()"
      onsubmit="FormEvent('submit')" value="form events"/>
      <!-- <br><br> -->

      <button type="submit">Submit</button>
      <p id="formStatus"> </p>
    </form>
  </div>

  <div class="event">
    <h3>Document/Window Event Handelling</h3>
    <p>Resize the window or reload/close the page.
    <a href="https://www.google.com">leave the page</a>
    </p>
    <div id="log"> </div>
  </div>

  <script>
    let changed= false
    function handleChange() {
      changed= true
      FormEvent('change')
    }

    function handleBlur() {
      if(!changed)
        FormEvent('blur')
      changed= false
    }

    function FormEvent(action){
```

```
        let info;
        if (action == 'focus')
            info = `<b>Event : onfocus</b><br>When the user focuses on an
element.`
        else if (action == 'blur')
            info = `<b>Event : onblur</b><br>When the focus is away from a form
element.`
        else if (action == 'input')
            info = `<b>Event : oninput</b><br>When the user is typing...`
        else if(action == 'change')
            info = `<b>Event : onchange</b><br>When the user modifies or changes
the value of a form element.`
        else if(action == 'submit')
            info = `<b>Event : onsubmit</b><br>When the user submits the form.`
        else
            info = `<b>Warning</b><br>No such event defined.`
        document.getElementById('formStatus').innerHTML= info;
    }

    const logDiv = document.getElementById('log')
    function log(message){
        const p= document.createElement('p');
        p.textContent= message
        logDiv.appendChild(p)
    }

    //load event
    window.onload= () => {
        log(' ✅ Window Loaded!')
    };

    window.onbeforeunload= ()=> {
        return 'are you sure to leave the page'
    };
```

```
        window.onresize = ()=> {
            log(`📏 Window resized: ${window.innerWidth} x
${window.innerHeight}` )
        };
        // window.addEventListener('load', () => log('✅ Window Loaded!'));
    </script>
</body>
</html>
```







Form Event Handelling

Field1: form events Submit

Event : onblur
When the focus is away from a form element.

Form Event Handelling

Field1: form Submit

Event : oninput
When the user is typing...

Form Event Handelling

Field1: form Submit

Event : onchange
When the user modifies or changes the value of a form element.

Form Event Handelling

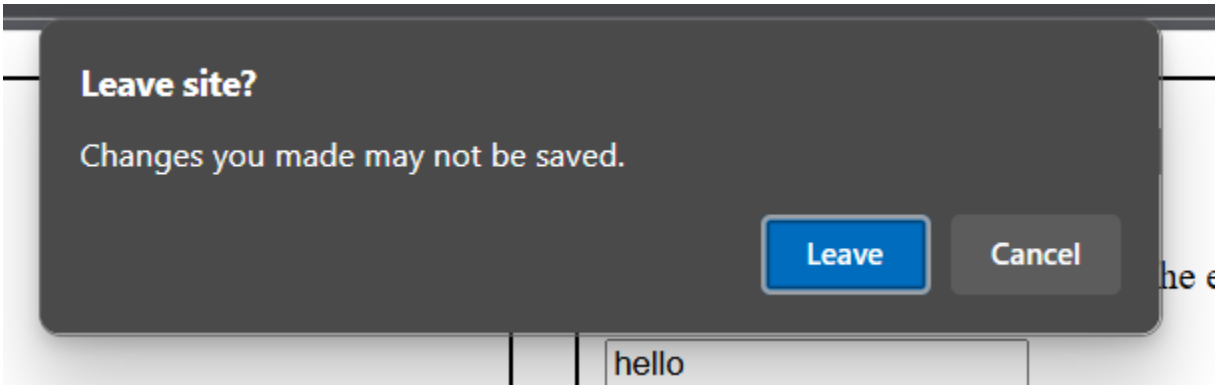
Field1: form Submit

Event : onsubmit
When the user submits the form.

Document/Window Event Handelling

Resize the window or reload/close the page. leave the page

- Window Loaded!
- Window resized: 768 x 651
- Window resized: 1528 x 738
- Window resized: 768 x 651
- Window resized: 768 x 651




```
Mouse Over                                  25.c_g.html:40
Clicked!                                    25.c_g.html:39
Mouse Over                                  25.c_g.html:40
mouse down                                  25.c_g.html:43
23 mouse move                               25.c_g.html:44
mouse out                                   25.c_g.html:41
Mouse Up                                    25.c_g.html:42
```

25.h

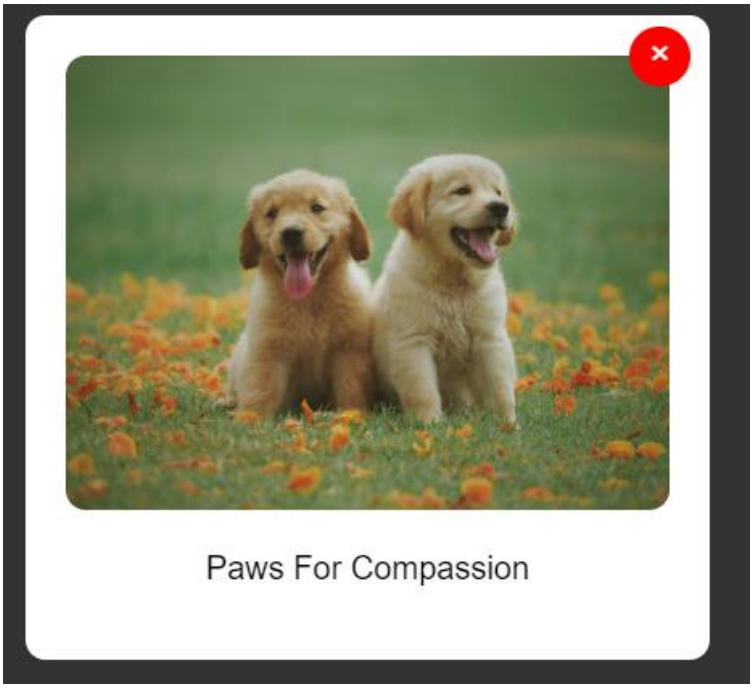
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Advertisement Popup</title>
    <style>
        body{
            font-family: Arial, Helvetica, sans-serif;
            margin: 0;
            padding: 0;
        }

        /* overlay for ad */
        #adOverlay {
            position: fixed;
            top: 0;
            left: 0;
            width: 100%;
            height: 100%;
            background-color: rgba(0, 0, 0, 0.8);
            display: flex;
            justify-content: center;
            align-items: center;
            z-index: 9999;
        }

        #adContainer {
            position: relative;
            background-color: white;
            padding: 20px;
            border-radius: 10px;
            text-align: center;
        }
```

```
        #adContainer img {
            width: 300px;
            max-width: 100%;
            height: auto;
            border-radius: 10px;
        }

        #closeBtn{
            position: absolute;
            top: 5px;
            right: 10px;
            font-size: 18px;
            background-color: red;
            color: white;
            border: none;
            border-radius: 50%;
            width: 30px;
            height: 30px;
            cursor: pointer;
        }

        /* hides content initially */
        #mainContent{
            display: none;
            padding: 20px;
        }
    </style>
</head>
<body>
    <!-- advertisement overlay -->
    <div id="adOverlay">
        <div id="adContainer">
            <button id="closeBtn">&times</button>     <!-- close button -->
```

```
        <img src="https://images.pexels.com/photos/1108099/pexels-photo-
1108099.jpeg" alt="Advertisement" height="250px">
        <p>Paws For Compassion</p>
      </div>
    </div>
    <!-- main page content  -->
     <div id="mainContent">
       <h1>Welcome to My Website</h1>
       <p>This is the main content of my webpage.</p>
     </div>

     <script>
       document.getElementById('closeBtn').addEventListener('click', () => {
          document.getElementById('adOverlay').style.display = 'none'
          document.getElementById('mainContent').style.display = 'block'
       })
     </script>

</body>
</html>
```

Welcome to My Website

This is the main content of my webpage.



25.i

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Highlighting Long Words</title>
    <style>
        .highlight{
            background-color: yellow;
        }
    </style>
</head>
<body>
    <h3>Paragraph</h3>
    <p id="p1">
        JavaScript is a powerful programming language commonly used in web
development to create interactive and dynamic user experiences.
    </p>

    <button onclick="highlightLongWords()">Highlight Long Words</button>

    <script>
        function highlightLongWords(){
            const para = document.getElementById('p1')
            const words= para.innerText.split(/\s+/)

            const highlighted= words.map(word => {
                if(word.length > 8){
                    return `<span class='highlight'>${word}</span>`
                }else{
                    return word
                }
            })
```

```
        para.innerHTML= highlighted.join(' ')
      }
   </script>

</body>
</html>
```

Paragraph

JavaScript is a powerful programming language commonly used in web development to create interactive and dynamic user experiences.

Highlight Long Words

Paragraph

JavaScript is a powerful programming language commonly used in web development to create interactive and dynamic user experiences.

Highlight Long Words

25.j

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Interactive Web Page</title>
   <style>
      body {
        font-family: Arial, sans-serif;
        padding: 20px;
        transition: background-color 0.3s ease;
      }

      .color-options {
        margin-bottom: 20px;
      }

      img {
        max-width: 300px;
        display: block;
        margin-bottom: 10px;
        cursor: pointer;
      }

      #description {
        display: none;
        font-style: italic;
      }
    </style>
</head>
<body>
   <h3>Change Background Color</h3>

   <div class="color-options">
```

```
      <label><input type="radio" name="bgColor" value="lightblue">Light
Blue</label>
      <label><input type="radio" name="bgColor" value="lightgreen">Light
Green</label>
      <label><input type="radio" name="bgColor" value="lightcoral">Light
Coral</label>
   </div>

   <h3>Hover to See Description</h3>
   <img id="dogImg" src="https://images.pexels.com/photos/1108099/pexels-
photo-1108099.jpeg">
   <p id="description">This is a sample image used for demonstration. Hovering
reveals this text.</p>

   <script>
      // background color change
      const radios = document.querySelectorAll("input[name='bgColor']")
      radios.forEach(radio => {
         radio.addEventListener('change', function (){
            document.body.style.backgroundColor = this.value
         })
      })

      // show/hide description on hover
      const image= document.getElementById('dogImg')
      const desc= document.getElementById('description')

      image.addEventListener('mouseover', () =>{
         desc.style.display = 'block'
      })

      image.addEventListener('mouseout', () =>{
         desc.style.display = 'none'
      })
```

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

Change Background Color

◉ Light Blue ○ Light Green ○ Light Coral

Hover to See Description

This is a sample image used for demonstration. Hovering reveals this text.

26.a

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DOM Node Properties</title>
    <style>
        #output {
            margin-top: 20px;
            font-family: monospace;
            white-space: pre-wrap;
            color: darkblue;
        }
    </style>
</head>
<body>
    <h2>DOM Node Properties</h2>

    <div id="container">
        <p>This is the <strong>first</strong> paragraph.</p>
        <p>This is the <em>second</em> paragraph.</p>
    </div>

    <button onclick="analyzeNode()">Analyze node</button>

    <div id="output"></div>

    <script>
        function analyzeNode(){
            const output= document.getElementById('output');

            const container= document.getElementById('container')
            const firstParagraph= container.firstElementChild
```

```
        let result= ""

        result += `nodeName: ${firstParagraph.nodeName}\n`
        result += `nodeValue ${firstParagraph.nodeValue}\n`  // usually null for
element nodes
        result += `nodeType: ${firstParagraph.nodeType}\n`  // 1 for
Element_node

        result += `parentNode.nodeName:
${firstParagraph.parentNode.nodeName}\n`

        result += `firstChild.nodeName:
${firstParagraph.firstChild.nodeName}\n`
        result += `firstChild.nodeValue
${firstParagraph.firstChild.nodeValue}\n`

        result += `lastChild.nodeName: ${firstParagraph.lastChild.nodeName}\n`
        result += `lastChild.nodeValue: ${firstParagraph.lastChild.nodeValue}\n`

        result += `previousSibling.nodeName: ${firstParagraph.previousSibling ?
firstParagraph.previousSibling.nodeName : 'null'}\n`
        result += `nextSibling.nodeValue ${firstParagraph.nextSibling ?
firstParagraph.nextSibling.nodeValue : 'null'}\n`

        result += `childNodes.length: ${firstParagraph.childNodes.length}\n`

        output.innerHTML = result
      }
   </script>

</body>
</html>
```

DOM Node Properties

This is the **first** paragraph.

This is the *second* paragraph.

Analyze node

```
nodeName: P
nodeValue null
nodeType: 1
parentNode.nodeName: DIV
firstChild.nodeName: #text
firstChild.nodeValue This is the
lastChild.nodeName: #text
lastChild.nodeValue:  paragraph.
previousSibling.nodeName: #text
nextSibling.nodeValue

childNodes.length: 3
```

26.b

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DOM Selection</title>
</head>
<body>
    <style>

    </style>
</body>
    <h2 id="main-title">Welcome to DOM Selection</h2>
    <p name="description">This is a paragraph selected by name.</p>
    <p name="description">Another paragraph with same name.</p>

    <div class="info">This is a div with class 'info'</div>
    <div class="info">Another div with class 'info'</div>

    <ul>
        <li>Item 1</li>
        <li class="special-item">Item 2</li>
        <li>Item 3</li>
    </ul>

    <button onclick="selectElements()">Select Elements</button>

    <script>
        function selectElements() {
            // 1. selecting by ID
            const byId = document.getElementById('main-title');
            console.log('By ID:', byId.innerText);

            // 2. Selecting by name
```

```
        const byName = document.getElementsByName('description');
        byName.forEach((el, i) => console.log(` By Name [${i}]: ` , el.innerText))

        // 3. Selecting by tag type
        const byTag = document.getElementsByTagName('li');
        for (let i=0; i < byTag.length; i++){
           console.log(` By Tag [li ${i}]: ` , byTag[i].innerText)
        }

        // 4. Selecting by CSS class
        const byClass = document.getElementsByClassName('info');
        Array.from(byClass).forEach((el, i) => console.log(` By class [info ${i}]: ` ,
el.innerText))

        // 5. Selecting by querySelector (returns first match only)
        const querySingle = document.querySelector('.special-item');
        console.log('By querySelector (.special-item): ',
querySingle.innerText)

        // 6. Selecting by querySelectorAll (returns NodeList of all matches)
        const queryAll = document.querySelectorAll('p[name="description"]');
        queryAll.forEach((el, i) => console.log(` By querySelectorAll
[p[name="description"] ${i}]: ` , el.innerText))
      }
   </script>

</html>
```

Welcome to DOM Selection

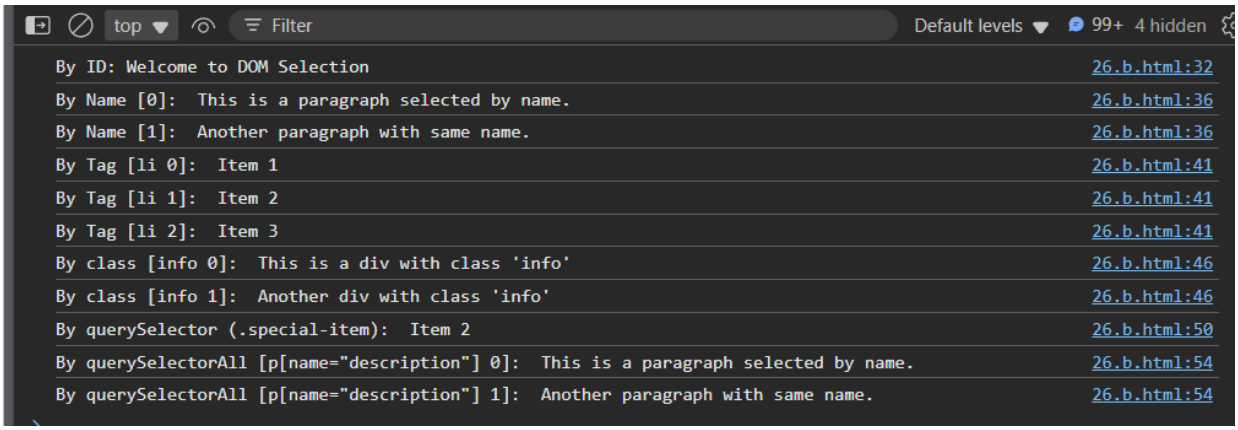
This is a paragraph selected by name.

Another paragraph with same name.

This is a div with class 'info'
Another div with class 'info'

- Item 1
- Item 2
- Item 3

Select Elements

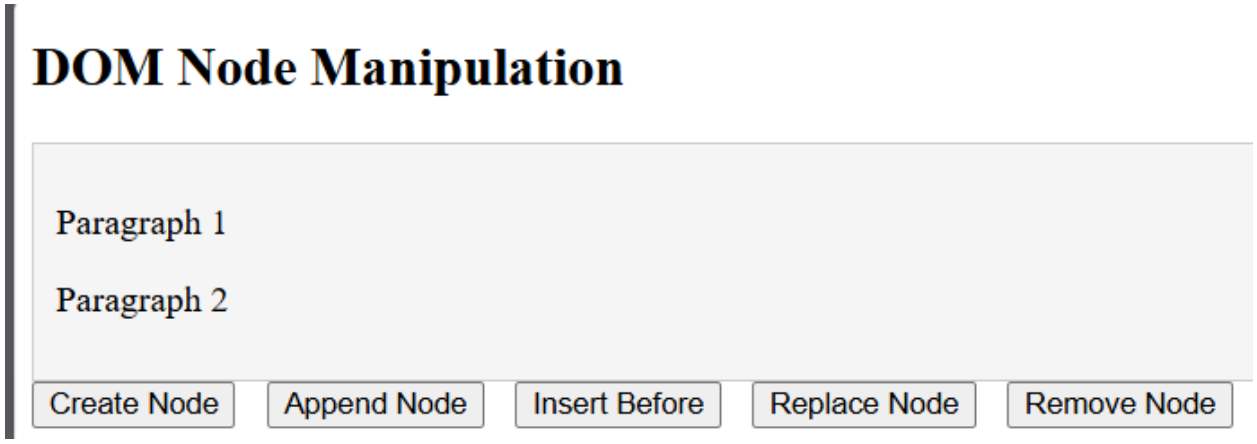


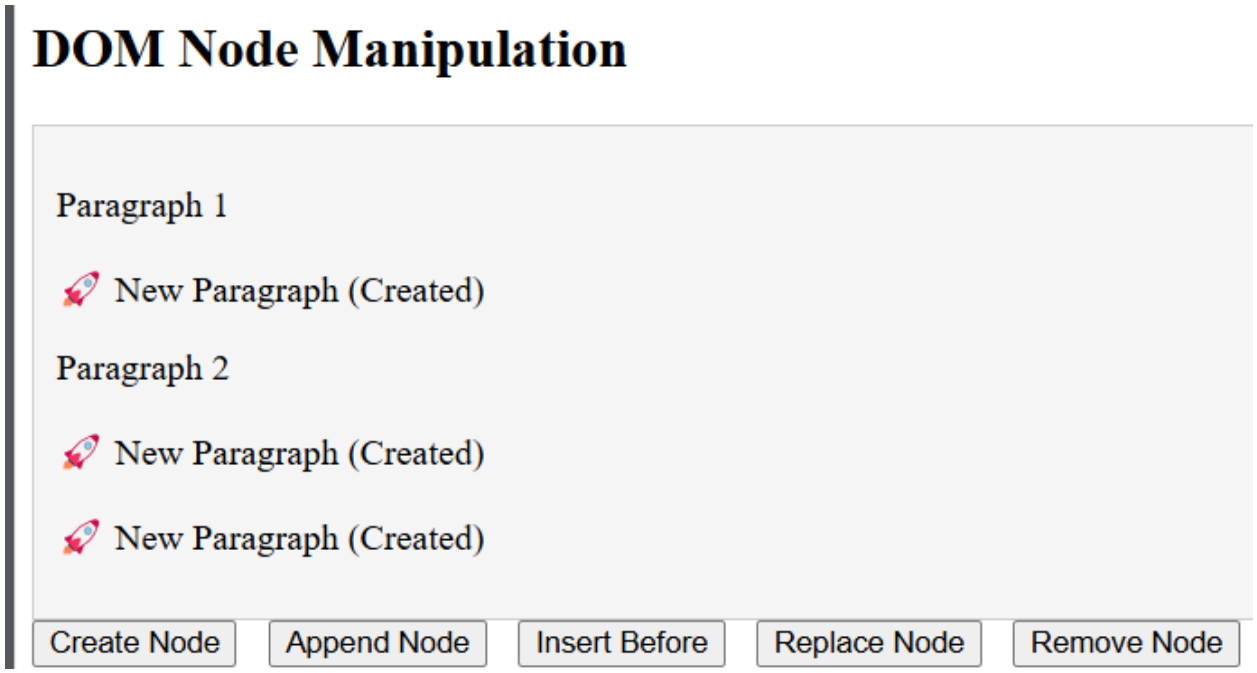
26.c

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>DOM Node Operations</title>
  <style>
    #output {
      border: 1px solid #ccc;
      padding: 10px;
      margin-top: 10px;
      background-color: #f5f5f5;
    }
    button {
      margin-right: 10px;
      margin-bottom: 10px;
    }
  </style>
</head>
<body>
  <h2>DOM Node Manipulation</h2>
  <div id="output">
    <p id="p1">Paragraph 1</p>
    <p id="p2">Paragraph 2</p>
  </div>

  <button onclick="createNode()">Create Node</button>
  <button onclick="appendNode()">Append Node</button>
  <button onclick="insertBeforeNode()">Insert Before</button>
  <button onclick="replaceNode()">Replace Node</button>
  <button onclick="removeNode()">Remove Node</button>

  <script>
    let newNode= null
```

```
  function createNode(){
    newNode= document.createElement('p'); // node creation
    newNode.textContent= '🚀 New Paragraph (Created)'
    alert('Node created! Now you can insert it.')
  }

  function appendNode(){
    if(newNode){
      document.getElementById('output').appendChild(newNode)
      alert('Node appended at the end!')
    }else{
      alert('Please create a node first!')
    }
  }

  function insertBeforeNode(){
    if(newNode){
      const refNode= document.getElementById('p2')
      document.getElementById('output').insertBefore(newNode, refNode) //
insert before refNode
    }else{
      alert('Please create a node first!')
    }
  }

  function replaceNode(){
    const oldNode= document.getElementById('p1')
    const replacement= document.createElement('p')
    replacement.textContent= '🔥 Replace paragraph'
    document.getElementById('output').replaceChild(replacement, oldNode)  //
node replacement
    alert('Paragraph 1 has been replaced!')
  }
```

```
  function removeNode(){
    const target= document.getElementById('p2')
    if(target){
      document.getElementById('output').removeChild(target)
      alert('Paragraph 2 has been removed!')
    }else{
      alert('Paragraph 2 already removed!')
    }
  }
 </script>
</body>
</html>
```





DOM Node Manipulation

🔥 Replace paragraph

🚀 New Paragraph (Created)

🚀 New Paragraph (Created)

🚀 New Paragraph (Created)

Create Node | Append Node | Insert Before | Replace Node | Remove Node

26.d

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>DOM Attribute Methods</title>
  <style>
    img{
      border: 2px solid #ccc;
      margin-bottom: 10px;
    }
    #message{
      margin-top: 10px;
      color: green;
    }
  </style>
</head>
<body>
  <h2>DOM Attribute Methods</h2>
  <img id="image" src="https://www.copahost.com/blog/wp-
content/uploads/2019/07/imgsize2-768x466.png" alt="Placeholder Image"
height="250px" width="400px">
  <br>

  <button onclick="checkAttribute()">Check 'alt' Attribute</button>
  <button onclick="getAttributeValue()">Get 'src' Attribute</button>
  <button onclick="setNewAttribute()">Set New 'title' Attribute</button>
  <button onclick="removeAltAttribute()">Remove 'alt' Attribute</button>

  <div id="message"></div>

  <script>
    const img= document.getElementById('image')
    const message= document.getElementById('message')
```

```
  function checkAttribute(){
    if(img.hasAttribute('alt')){
      message.textContent= '✅ The image has an "alt" attribute.'
    }else {
      message.textContent= '❌ The image has no "alt" attribute.'
    }
  }

  function getAttributeValue(){
    const srcValue= img.getAttribute('src')
    message.textContent= `🔍 Image source is: ${srcValue}`
  }

  function setNewAttribute(){
    img.setAttribute('title', 'This is a placeholder image.')
    message.textContent= '🎯 "title" attribute set successfully.'
  }

  function removeAltAttribute(){
    img.removeAttribute('alt')
    message.textContent= '⛔ "alt" attribute removed!'
  }
 </script>

</body>
</html>
```

DOM Attribute Methods

Check 'alt' Attribute | Get 'src' Attribute | Set New 'title' Attribute | Remove 'alt' Attribute

🎯 "title" attribute set successfully.

26.e

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add Book Record</title>
    <style>
        body{
            font-family: Arial, Helvetica, sans-serif;
            margin: 20px;
        }
        .form-container{
            margin-bottom: 20px;
        }
        .form-container button{
            margin-left: 10px;
        }
        table{
            border-collapse: collapse;
            width: 100%;
            max-width: 800px;
            margin-top: 20px;
        }
        table, th, td{
            border: 1px solid black;
        }
        th, td{
            padding: 8px;
            text-align: left;
        }
        tbody tr:hover{
            background-color: green;
            color: white;
        }
```

```
    </style>
</head>
<body>
    <h3>Add Book Records</h3>
        <div class="form-container">
            <label>Book Id:
                <input type="text" id="bookId" value="100" readonly>
            </label>
            <label>Title:
                <input type="text" id="title">
            </label>

            <label>Author(s):
                <input type="text" id="authors">
            </label>

            <label>Price:
                <input type="number" step="0.1" id="price">
            </label>
            <button onclick="incrementBookId()">+</button>
        </div>
        <button onclick="saveBook()">Save</button>

        <table id="bookTable">
            <thead>
                <tr>
                    <th>Book Id</th>
                    <th>Title</th>
                    <th>Author(s)</th>
                    <th>Price</th>
                </tr>
            </thead>
            <tbody id="bookTableBody"></tbody>
        </table>
```

```
<script>
    let bookId = 100;
    function incrementBookId(){
        bookId ++
        document.getElementById('bookId').value = bookId
        // add a new empty row to the table
        const tableBody= document.getElementById('bookTableBody')
        const newRow = tableBody.insertRow()
        newRow.insertCell(0).innerText= bookId
        newRow.insertCell(1).innerText= ''
        newRow.insertCell(2).innerText= ''
        newRow.insertCell(3).innerText= ''
        // add click event to the new row
        newRow.addEventListener('click', function (){
            populateForm(newRow)
        })
    }
    function saveBook(){
        const title= document.getElementById('title').value
        const authors= document.getElementById('authors').value
        const price= document.getElementById('price').value

        if(title && authors && price){
            const tableBody= document.getElementById('bookTableBody')
            const rows= tableBody.getElementsByTagName('tr')

            // update the row with the entered values
            for(let i=0; i<rows.length; i++){
                if(rows[i].cells[0].innerText == bookId){
                    rows[i].cells[1].innerText = title
                    rows[i].cells[2].innerText = authors
                    rows[i].cells[3].innerText = price
                    break
                }
```

```
            }

            // clear the input fields (except Book Id)
            document.getElementById('title').value= ''
            document.getElementById('authors').value= ''
            document.getElementById('price').value= ''
         }else{
            alert('Please fill in all fields before saving.')
         }
      }

      function populateForm(row){
         // populate the form fields with the row data
         document.getElementById('bookId').value = row.cells[0].innerText
         document.getElementById('title').value = row.cells[1].innerText
         document.getElementById('authors').value = row.cells[2].innerText
         document.getElementById('price').value = row.cells[3].innerText

         // update the global bookId variable
         bookId= parseInt(row.cells[0].innerText, 10) // radix 10 decimal
      }

      // add click event to existing rows (if any)
      document.addEventListener('DOMContentLoaded', function (){
         const rows=
document.getElementById('bookTableBody').getElementsByTagName('tr')
         for(let i=0; i< rows.length; i++){
            rows[i].addEventListener('click', function (){
               populateForm(rows[i])
            })
         }
      } )

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

Add Book Records

Book Id: 102 Title: Mahabir Pun Atmakatha Author(s): Mahabir Pun Price: 599.5 +

Save

Book Id	Title	Author(s)	Price
101	Siris ko Phul	Parijat	345.0
102	Mahabir Pun Atmakatha	Mahabir Pun	599.5
103	MunaMadan	Laxmi Prasad Devkota	350

26.f

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Display Districts Dynamically</title>
  <style>
    ul {
      padding: 0;
      list-style-type: none;
    }
    li {
      background-color: #f5f5f5;
      padding: 8px;
      margin: 5px 0;
      border-radius: 4px;
    }
    #message {
      margin-top: 20px;
      color: green;
    }
  </style>
</head>

<body>
  <h3>Districts in Nepal</h3>
  <button onclick="displayDistricts()">Display Districts</button>

  <ul id="districtList"></ul>
  <div id="message"></div>

  <script>
    // create a js array with some districts
```

```
   const districts= ['Kathmandu', 'Chitwan', 'Lalitpur', 'Pokhara', 'Chitwan', 'Biratnagar', 'Dharan']

   // display the districts dynamically in an unordered list using DOM Manipulation
   function displayDistricts(){
     const ul= document.getElementById('districtList')
     ul.innerHTML= ''   // clear existing list items (if function is called multiple times)
     // loop through the districts array and create list items dynamically
     districts.forEach(district => {
       const li= document.createElement('li')
       li.textContent= district
       ul.appendChild(li)
     })
     // provide feedback to the use
     document.getElementById('message').textContent= '✅ Districts displayed successfully!'
   } </script></body></html>
```

Districts in Nepal

Display Districts

- Kathmandu
- Chitwan
- Lalitpur
- Pokhara
- Chitwan
- Biratnagar
- Dharan

✅ Districts displayed successfully!

27.a

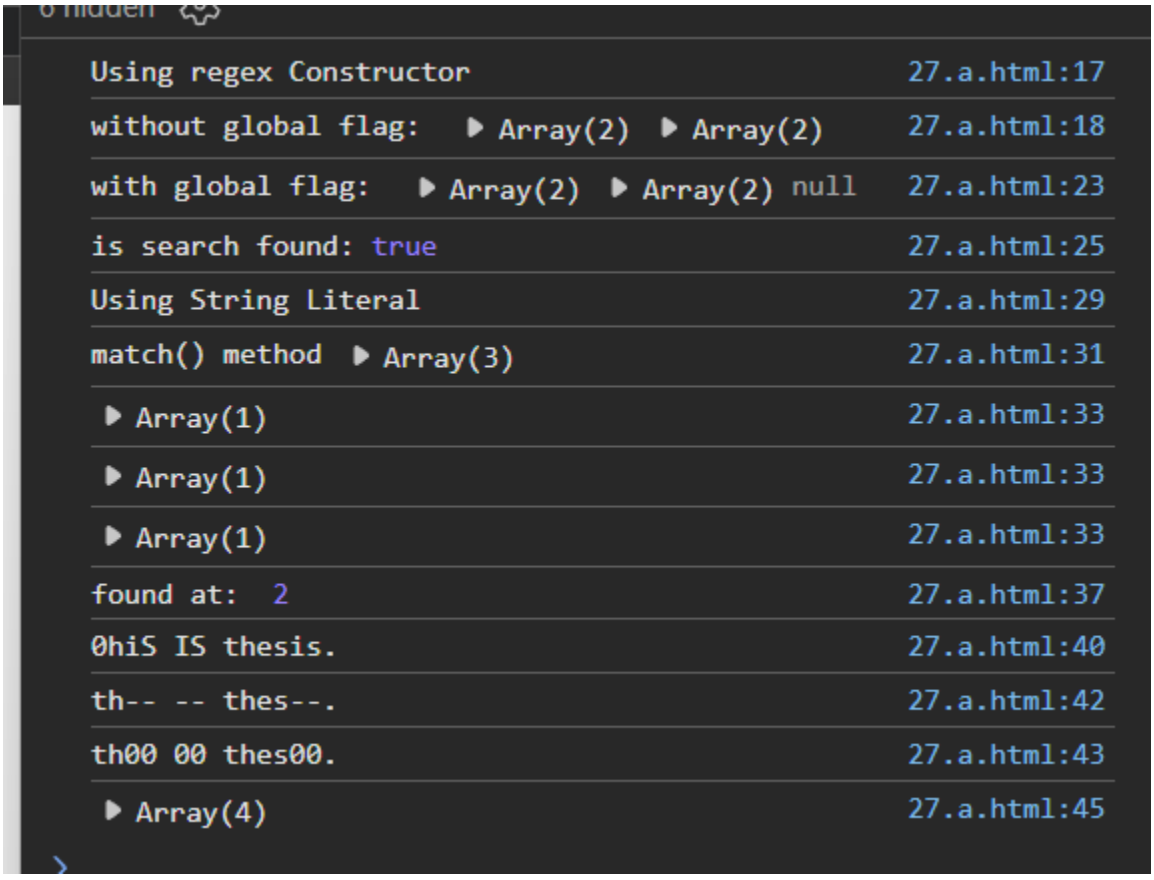
```
<!DOCTYPE html>
<html lang="en">
<body>
   <h4>Ways of creating Regular Expression</h4>
   <p>1. Using the RegExp constructor new RegExp() <br>2. Using the literal
notation <strong>/pattern/</strong> </p>
   <p>check console for regular expression in work.</p>

   <script>
      let pattern= '(a.c)d'
      let regex= new RegExp(pattern, 'i')
      let text= 'eAxcdbraycdew'
      console.log('Using regex Constructor')
      console.log('without global flag: ',
         regex.exec(text), regex.exec(text))

      // with global flag
      regex= new RegExp(pattern, 'gi')
      console.log('with global flag: ',
         regex.exec(text), regex.exec(text), regex.exec(text))
      console.log('is search found:' ,regex.test(text))

      text= 'thiS IS thesis.
      console.log('Using String Literal')
      console.log('match() method',text.match(/is/gi))
      for (item of text.matchAll(/is/gi)){
         console.log(item)
      }
      // using search() on string literal
      console.log('found at: ', text.search(/is/i))

      // using replace and replaceAll method
      console.log(text.replace(/[a-z]/, '0'))
      // search with global flag(/g) is similar to replaceAll()
```

```
      console.log(text.replace(/is/gi, '--'))
      console.log(text.replaceAll(/is/gi, '00'))

      console.log('1-22-333-4444'.split(/-/))

   </script>

</body>
</html>
```

Ways of creating Regular Expression

1. Using the RegExp constructor new RegExp()
2. Using the literal notation **/pattern/**

check console for regular expression in work.



27.b

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        input{
            margin-left: 20px;
        }
    </style>
</head>
<body>
    <h3>Regular Expression Pattern</h3>
    1. The first character of a string must be uppercase. <br>
    <input id="field1" type="text" onchange="checkRegex(this, '^[A-Z]',
'status1')">
    <span id="status1"></span><br>

    2. A given value must contain a question marks, a dash and an underscore.<br>
    <input id="field2" type="text" onchange="checkRegex(this, '^(?=.*\?)(?=.*-
)(?=.*_).+$', 'status2')">
    <span id="status2"></span><br>

    3. The value must start from A or a and must end with full stop (.) or
exclamation (!). <br>
    <input id="field3" type="text" onchange="checkRegex(this, '^[aA].*[.!]$',
'status3')">
    <span id="status3"></span><br>

    4. The value must contain only one blank space. <br>
    <input id="field4" type="text" onchange="checkRegex(this, '^\\S+\\s\\S+$',
'status4')">
```

```
   <span id="status4"></span><br>

   5. The string must not contain any digits. <br>
   <input id="field5" type="text" onchange="checkRegex(this, '^[^0-9]*$',
'status5')">
   <span id="status5"></span><br>

   6. The string must conation at least a symbol from the group of symbol ( , , @,
$, _, %,
   !, &, *, ^, ~, -, +, (, ), {,}, [, ], . , : )<br>
   <!-- Escape character: in RegExp() constructor  double backlash \\ whereas in
string literal single \  -->
   <input id="field6" type="text" onchange="checkRegex(this,
'^.*[,@$_%!&*^~\\\-T(){}[\\].:].*$', 'status6')">
   <span id="status6"></span><br>

   7. The value must not contain any special character and digits. <br>
   <input id="field7" type="text" onchange="checkRegex(this, '^[a-zA-Z\\s]+$',
'status7')">
   <span id="status7"></span><br>

   8. The value must contain @ symbol between two words each of at least one
character. <br>
   <input id="field8" type="text" onchange="checkRegex(this, '^\\w+@\\w+$',
'status8')">
   <span id="status8"></span><br>

   9. The string must not start with digit. <br>
   <input id="field9" type="text" onchange="checkRegex(this, '^\\D.*$',
'status9')">
   <span id="status9"></span><br>

   10. The string must start with underscore character or alphabets. <br>
   <input id="field10" type="text" onchange="checkRegex(this, '^[_a-zA-Z].*$',
'status10')">
```

```
<span id="status10"></span><br>

11. The value must start with 'BCA-TU-' and after these character there must be only
digits.<br>
<input id="field11" type="text" onchange="checkRegex(this, '^(BCA-TU-)[0-9]*$', 'status11')">
<span id="status11"></span><br>

12. The value must be in email address format.  <br>
<input id="field12" type="text" onchange="checkRegex(this, '^[a-zA-Z0-9._%+-]+@[a-zA-Z]+(\.[a-zA-Z]{2,})+$', 'status12')">
<span id="status12"></span><br>

13. The value must contain at least a vowel letter. <br>
<input id="field13" type="text" onchange="checkRegex(this, '.*[aeiouAEIOU].*', 'status13')">
<span id="status13"></span><br>

14. The value contain ten digits and first digit cannot be 0.  <br>
<input id="field14" type="text" onchange="checkRegex(this, '^[1-9][0-9]{9}$', 'status14')">
<span id="status14"></span><br>

15. The value should be email address and its must end with 'edu.np'. <br>
<input id="field15" type="text" onchange="checkRegex(this, '^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+(\.edu.np)$', 'status15')"">
<span id="status15"></span><br>

16. The value should be email address and it must contain 'gmail' or 'yahoo' after @
symbol.   <br>
<input id="field16" type="text" onchange="checkRegex(this, '^[a-zA-Z0-9._%+-]+@(gmail|yahoo)(\.[a-zA-Z]{2,})+$', 'status16')">
<span id="status16"></span><br>
```

```
  17. The regex must match if the string contains 'nepal' case insensitively.<br>
  <input id="field17" type="text" onchange="checkRegex(this,
'([Nn][Ee][Pp][Aa][Ll])', 'status17')">
  <span id="status17"></span><br>
  <!-- can be used
    console.log('nepal'.match(/nePaL/i))
    also if defined using string literal with flags or
    new RegExp(pattern,flag) -->


  <script>
     function checkRegex(inputElement, pattern, statusElementId){
        const regex= new RegExp(pattern)
        const effectText= document.getElementById(statusElementId)

        if (!regex.test(inputElement.value)){
           inputElement.style.outline = '2px solid red';
           effectText.textContent = ' ⚠️ Warning!!!';
           effectText.style.color = 'red';
        }else{
           inputElement.removeAttribute('style')
           effectText.textContent= ' ✅ Good'
           effectText.removeAttribute('style')
        }
     }
  </script>

</body>
</html>
```

Regular Expression Pattern

1. The first character of a string must be uppercase.
   Hello ✅ Good
2. A given value must contain a question marks, a dash and an underscore.
   what about you? _g--d? ✅ Good
3. The value must start from A or a and must end with full stop (.) or exclamation (!).
   A sky full of stars. ✅ Good
4. The value must contain only one blank space.
   do it ✅ Good
5. The string must not contain any digits.
   no digit🤩 ✅ Good
6. The string must conation at least a symbol from the group of symbol (, , @, $, _, %, !, &, *, ^, ~, -, +, (,), {,}, [,], . , :)
   ~!@#$%^&*()_+[]{} done ✅ Good
7. The value must not contain any special character and digits.
   no digits ✅ Good
8. The value must contain @ symbol between two words each of at least one character.
   me@you ✅ Good
9. The string must not start with digit.
   12wrong11 ⚠️ Warning!!!
10. The string must start with underscore character or alphabets.
    _myself ✅ Good
11. The value must start with 'BCA-TU-' and after these character there must be only digits.
    BCA-TU-124 ✅ Good
12. The value must be in email address format.
    valid.23@js.com ✅ Good
13. The value must contain at least a vowel letter.
    one ✅ Good
14. The value contain ten digits and first digit cannot be 0.
    9123456789 ✅ Good
15. The value should be email address and its must end with 'edu.np'.
    test.400@tu.edu.np ✅ Good
16. The value should be email address and it must contain 'gmail' or 'yahoo' after @ symbol.
    your@yahoo.com.gov.uk ✅ Good
17. The regex must match if the string contains 'nepal' case insensitively.
    NePaL ✅ Good

28.a

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Objects</title>
</head>
<body>
    <h3>JS document.forms</h3>
    <form name="myForm">
        <label>Name:
            <input type="text" name= 'name' id="username">
        </label>
        <label>Email:
            <input type="email" name="email" id="email">
        </label>
        <button type="button" onclick="displayFormDate()">Submit</button>
    </form>
    <div id="output"></div>
    <script>
        function displayFormDate(){
            // access form using doucment.forms
            let form= document.forms['myForm']

            // get values from the form fields
            let name= form['username'].value;
            let email= form['email'].value
            // display
            document.getElementById('output').innerHTML=
            'Name: ' + name + '<br>Email: ' + email;
        }
    </script>
</body>
</html>
```

JS document.forms

Name: Sushanka Email: its.secret@gmail.com Submit

Name: Sushanka
Email: its.secret@gmail.com

28.b.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Registration Form</title>
    <link rel="stylesheet" href="28.b.css">
    <script defer src="28.b.js"></script>
</head>
<body>
    <h3>Registration Form</h3>
    <!--With POST, form data is sent in the HTTP body, not visible in the URL.
    GET method appends form data to the URL as query parameters (?key=value
format). Sensitive data being exposed in the URL (like passwords ✗) -->
    <form id="registrationForm" method="post">
        <label>First Name*:
            <input type="text" name="fname" id="fname">
        </label>
        <span class="error" id="fnameError"></span><br>

        <label>Last Name:
            <input type="text" name="lname" id="lname">
        </label>
        <span class="error" id="lnameError"></span><br>

        <label>Date of Birth*:
            <input type="date" name="dob" id="dob">
        </label>
        <span class="error" id="dobError"></span><br>

        <label>Gender*:
            <select id="gender">
                <option value="">----Select----</option>
                <option>Male</option>
```

```
        <option>Female</option>
        <option>Other</option>
    </select>
</label>
<span class="error" id="genderError"></span><br>

<label>Province*:
    <select id="province">
        <option value="">----Select----</option>
        <option>Province 1</option>
        <option>Province 2</option>
        <option>Bagmati</option>
        <option>Gandaki</option>
        <option>Lumbini</option>
        <option>Karnali</option>
        <option>Sudurpashchim</option>
    </select>
</label>
<span class="error" id="provinceError"></span><br>

<label>Cell Phone*:
    <input type="number" name="phone" id="phone">
</label>
<span class="error" id="phoneError"></span><br>

<label>Email:
    <input type="text" name="email" id="email">
</label>
<span class="error" id="emailError"></span><br>

<label>User Name*: <input type="text" id="username"></label>
<span class="error" id="usernameError"></span><br>

<label>Password*: <input type="password" id="password"></label>
```

```
      <span class="error" id="passwordError"></span><br>

      <label>Confirm Password*:
         <input type="password" name="confirmPass" id="confirmPass">
      </label>
      <span class="error" id="confirmPassError"></span><br>

      <button type="submit">Register</button>
   </form>

</body>
</html>
```

28.b.css

```
body {
    font-family: Arial, sans-serif;
    padding: 20px;
  }

  label {
    display: block;
    margin-top: 10px;
  }

  .error {
    color: red;
    font-size: 12px;
    margin-left: 10px;
  }
```

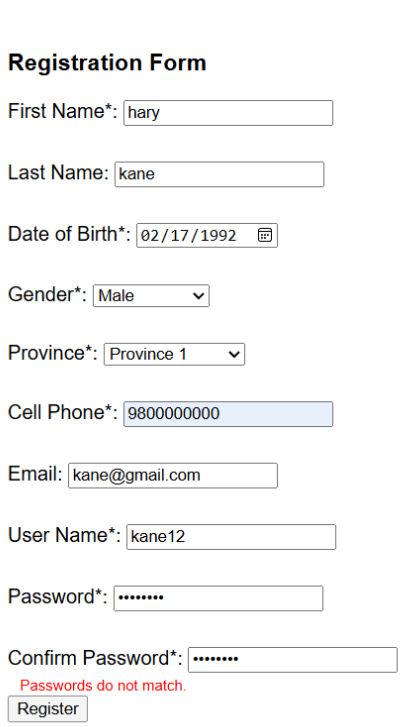
28.b.js

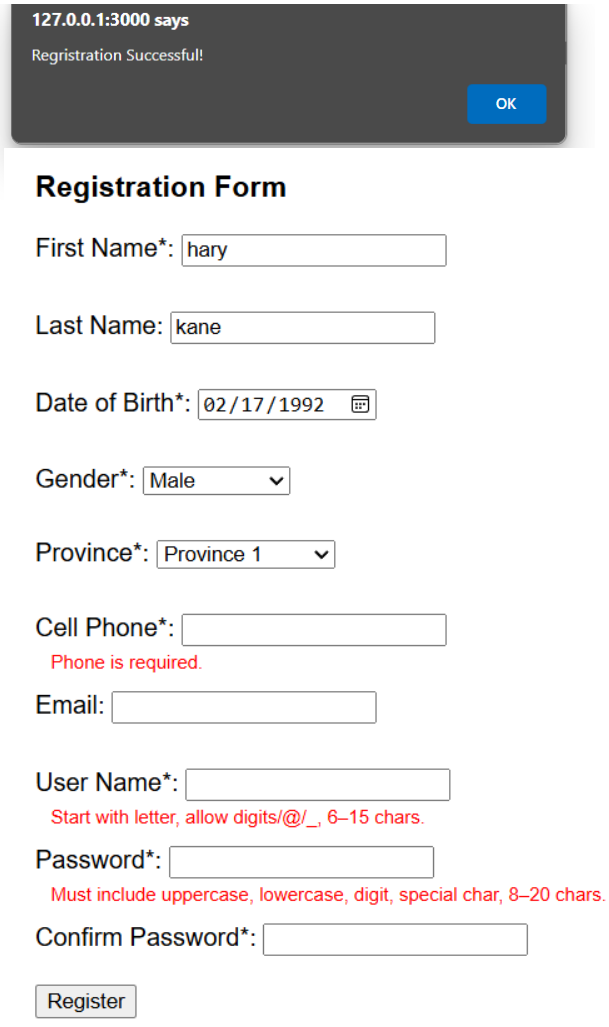
```
document.getElementById('registrationForm').addEventListener('submit',
function(e) {
   e.preventDefault()

   let valid= true;

   const showError= (id, message) => {
      document.getElementById(id).textContent= message
      valid= false
   }

   const clearError= (id)=>{
      document.getElementById(id).textContent= ''

   }

   const nameRegex= /^[a-zA-Z]{1,50}$/
   const phoneRegex= /^9[7-8][0-9]{8}$/
   const emailRegex= /^[^\s@]+@[^\s@]+\.[^\s@]+$/
   const usernameRegex = /^[A-Za-z][A-Za-z0-9@_]{5,14}$/;
   const passwordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[^A-Za-z0-
9]).{8,20}$/;

   const fname= document.getElementById('fname').value.trim()
   const lname= document.getElementById('lname').value.trim()
   const dob= document.getElementById('dob').value
   const gender= document.getElementById('gender').value
   const province= document.getElementById('province').value
   const phone= document.getElementById('phone').value.trim()
   const email= document.getElementById('email').value.trim()
   const username= document.getElementById('username').value.trim()
   const password= document.getElementById('password').value
   const confirmPass= document.getElementById('confirmPass').value
```

```
    //first name
    if(!fname || !nameRegex.test(fname)){
        showError('fnameError', 'Required. Only alphabets upto 50 characters
long.')
    }else{
        clearError('fnameError')
    }

    // last name (optional)
    if(lname && !nameRegex.test(lname)){
        showError('lnameError', 'Only alphabets upto 50 characters long.')
    }else{
        clearError('lnameError')
    }

    // dob
    if(!dob){
        showError('dobError', 'Date of Birth is required.')
    }else{
        clearError('dobError')
    }

    // Gender
    if(!gender){
        showError('genderError', 'Gender is required.')
    }else{
        clearError('genderError')
    }

    // province
    if(!province){
        showError('provinceError', 'Province is required.')
    }else{
        clearError('provinceError')
    }
```

```
    // Phone
    if(!phoneRegex.test(phone)){
        showError('phoneError', 'Phone is required.')
    }else{
        clearError('phoneError')
    }

    // email(optional)
    if(email && !emailRegex.test(email)){
        showError('emailError', 'Invalid Email format!')
    }else{
        clearError('emailError')
    }

    // username
    if(!username || !usernameRegex.test(username)){
        showError('usernameError', 'Start with letter, allow digits/@/_, 6-15
chars.')
    }else{
        clearError('usernameError')
    }

    // password
    if(!password || !passwordRegex.test(password)){
        showError('passwordError', 'Must include uppercase, lowercase, digit,
special char, 8-20 chars.')
    }else{
        clearError('passwordError')
    }

    // confirm passowrd
    if(confirmPass !== password){
        showError('confirmPassError', 'Passwords do not match.')
    }else{
        clearError('confirmPassError')
```

```
  }

  if (valid){
     alert('Regristration Successful!')
     this.submit()  // (optional)
  }

})
```





29

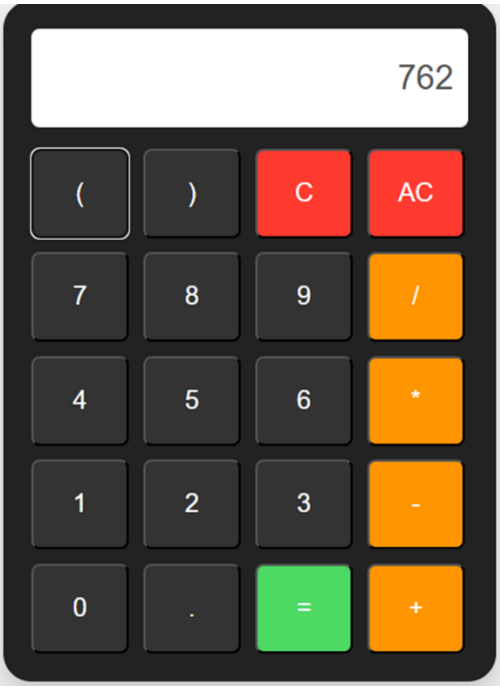
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calculator App</title>
    <style>
        body{
            display: flex;
            justify-content: center;
            align-items: center;
            border: 2px solid;
            height: 100vh;

            background: #f3f3f3;
        }
        #calculator{
            background: #222;
            padding: 20px;
            border-radius: 20px;
            box-shadow: 0 8px 20px rgba(0,0,0,0.2);
        }

        #display{
            /* width: 100%; */
            height: 50px;
            background: #fff;
            border: none;
            margin-bottom: 15px;
            font-size: 24px;
            padding: 10px;
            border-radius: 6px;
            text-align: right;
        }
```

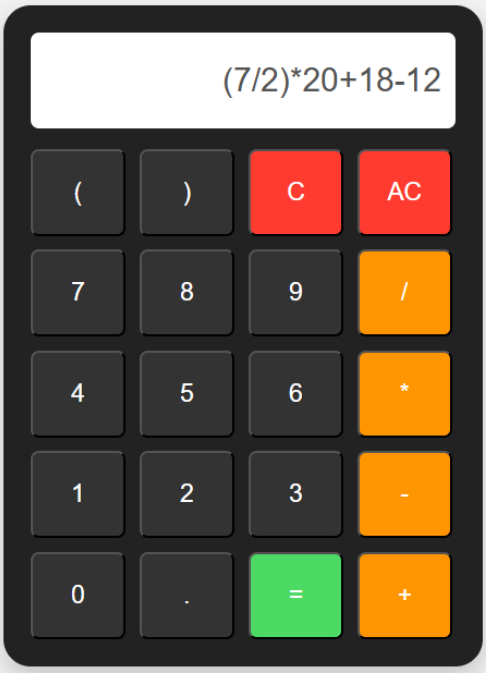
```
#buttons{
    display: grid;
    grid-template-columns: repeat(4, 70px);
    gap: 10px;

}

button{
    padding: 20px;
    font-size: 18px;
    border-radius: 6px;
    cursor: pointer;
    background-color: #333;
    color: #fff;
    /* transition: 0.8 ease; */
}

button:hover{
    background-color: #555;
}

.operator{
    background: #ff9500;
}
.operator:hover{
    background: #e08900;
}

#equal{
    background: #4cd964;
}
#equal:hover{
    background: #3bc95a;
}
```

```
        .clear {
            background: #ff3b30;
        }

        .clear:hover {
            background: #d93027;
        }
    </style>
</head>
<body>
    <div id="calculator">
        <input type="text" id="display" disabled>
        <div id="buttons">
            <button onclick="append('(')">(</button>
            <button onclick="append(')')">)</button>
            <button class="clear" onclick="clearDisplay()">C</button>
            <button class="clear" onclick="allClear()">AC</button>
            <!-- <br> -->
            <button onclick="append(7)">7</button>
            <button onclick="append(8)">8</button>
            <button onclick="append(9)">9</button>
            <button class="operator" onclick="append('/')">/</button>
            <!-- <br> -->
            <button onclick="append(4)">4</button>
            <button onclick="append(5)">5</button>
            <button onclick="append(6)">6</button>
            <button class="operator" onclick="append('*')">*</button>
            <!-- <br> -->
            <button onclick="append(1)">1</button>
            <button onclick="append(2)">2</button>
            <button onclick="append(3)">3</button>
            <button class="operator" onclick="append('-')">-</button>
            <!-- <br> -->
            <button onclick="append(0)">0</button>
```

```
        <button onclick="append('.')">.</button>
        <button id="equal" onclick="calculate()">=</button>
        <button class="operator" onclick="append('+')">+</button>
    </div>
</div>

<script>
    const display= document.getElementById('display')
    function append(value){
        if (display.value === 'Error' || display.value == '0') {
            display.value = ''; // clear previous error
        }
        display.value += value
    }

    function clearDisplay() {
        if ((display.value === 'Error') || (display.value.length == 1)){
            allClear()
        }else{
            display.value= display.value.slice(0,-1)
        }
    }

    function allClear(){
        display.value= '0'
    }

    function calculate(){
        try{
            display.value= eval(display.value)
            // display.value=
(Math.round(eval(display.value)*10**10)/(10**10)).toFixed(8)
        }
        catch{
            display.value= 'Error'
```

```
        }
    }

    // handle keyboard input
    document.addEventListener('keydown', (e)=>{
        const key = e.key
        if(!isNaN(key) || ['+','-','*','/','.','(',')'].includes(key))
            append(key);

        else if(key === 'Enter'){
            calculate();
        }
        else if(key === 'Backspace') clearDisplay()
        else if(key.toLowerCase() === 'c') allClear()

    });

    display.value= 0
  </script>
</body>
</html>
```





30

```
<!DOCTYPE html>
<html>
<head>
   <title>JavaScript Cookies Demo</title>
</head>
<body>
   <h2>JavaScript Cookie Operations</h2>
   <button onclick="createCookies()">Create Cookies</button>
   <button onclick="readCookies()">Read Cookies</button>
   <button onclick="changeCookie()">Change Cookie</button>
   <button onclick="deleteCookie()">Delete Cookie</button>

   <pre id="output" style="background: #f0f0f0; padding: 10px;"></pre>

   <script>
      // Function to create cookies in different ways
      function createCookies() {
         document.cookie = "username=Sushanka Khadka; path=/"; // Simple
session cookie
         document.cookie = "role=admin; max-age=3600; path=/"; // Persistent
cookie for 1 hour
         document.cookie = "theme=dark; path=/; expires=" + new Date(Date.now()
+ 86400000).toUTCString(); // Cookie with expiry and path

         showOutput("Cookies created!");
      }

      // Function to read cookies
      function readCookies() {
         const cookies = document.cookie;
         showOutput("Current cookies:\n" + cookies);
      }
```

```
        // Function to change cookie
        function changeCookie() {
            document.cookie = "username=Unknown; path=/"; // Overwrites the
previous value
            showOutput("Cookie 'username' updated to Unknown.");
        }

        // Function to delete cookie
        function deleteCookie() {
            // To delete, set the cookie with an expired date
            document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 UTC;
path=/;";
            document.cookie = "role=; expires=Thu, 01 Jan 1970 00:00:00 UTC;
path=/;";
            document.cookie = "theme=; expires=Thu, 01 Jan 1970 00:00:00 UTC;
path=/;";

            showOutput("Cookies deleted.");
        }

        // Helper function to print to the page
        function showOutput(msg) {
            document.getElementById("output").textContent = msg;
        }
    </script>
</body>
</html>
```

JavaScript Cookie Operations

Create Cookies | Read Cookies | Change Cookie | Delete Cookie

```
Current cookies:
username=Sushanka Khadka; role=admin; theme=dark
```

JavaScript Cookie Operations

Create Cookies | Read Cookies | Change Cookie | Delete Cookie

```
Cookie 'username' updated to Unknown.
```

JavaScript Cookie Operations

Create Cookies | Read Cookies | Change Cookie | Delete Cookie

```
Cookies deleted.
```

JavaScript Cookie Operations

Create Cookies | Read Cookies | Change Cookie | Delete Cookie

```
Current cookies:
```

31. a-b

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>jQuery demo</title>
   <script
src="https://cdn.jsdelivr.net/npm/jquery@3.7.1/dist/jquery.min.js"></script>
   <style>
      body { font-family: Arial; padding: 20px; }
      label { display: block; margin-top: 10px; }
      .error { color: red; font-size: 0.9em; }
      input, select { margin-top: 5px; }
   </style>
</head>
<body>
   <h2>Signup Form</h2>
   <form id="signupForm">
      <label>Name:
         <input type="text" id="name">
         <span class="error" id="nameError"></span>
      </label>

      <label>Email:
         <input type="text" id="email">
         <span class="error" id="emailError"></span>
      </label>

      <label>Password:
         <input type="password" id="password">
         <button type="button" id="togglePassword">Show</button>
         <span class="error" id="passwordError"></span>
      </label>
```

```
    <label>Gender:
        <input name="gender" type="radio"> Male
        <input name="gender" type="radio"> Female
        <span class="error" id="genderError"></span>
    </label>

    <label>Interests:
        <input name="interests" type="checkbox" value="sports"> Sports
        <input name="interests" type="checkbox" value="study"> Study
        <input name="interests" type="checkbox" value="music"> Music
        <span class="error" id="interestError"></span>
    </label>

    <label>Country:
        <select id="country">
            <option value="">---select---</option>
            <option value="Nepal">Nepal</option>
            <option value="Others">Others</option>
        </select>
        <span class="error" id="countryError"></span>
    </label><br>

    <button type="submit">Submit</button>
</form>


<script>
    // prevents jQuery from loading before document elements
    // // $(function(){       shortcut to $(document.ready(function(){}))

    $(document).ready(function(){
        // toggle passowrd button
        $('#togglePassword').click(function() {
                const passwordField= $('#password');
```

```
            const type= passwordField.attr('type') === 'password' ?'text' :
'password';
            passwordField.attr('type', type);

            // change this text based on the current state
            $(this).text(type === 'password' ? 'Show' : 'Hide');
        })

        $('#signupForm').submit(function(event){
            event.preventDefault();  // prevent form
form

            let valid =true;

            // clear previous errors
            $('.error').text('');

            //Name
            if($('#name').val().trim() === ''){
                $('#nameError').text('Name is required')
                valid= false;
            }

            //Email
            const email= $('#email').val().trim();
            const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
            if(email === ''){
                $('#emailError').text('Email is required');
                valid= false;
            }else if (!emailPattern.test(email)) {
                $('#emailError').text('Enter a valid Email');
                valid= false;
            }

            //Password
```

```
            if ($('#password').val().length < 6){
                $('#passwordError').text('Password must be at least 6
characters');
                valid= false;
            }

            //Gender
            if(!$("input[name='gender']:checked").val()){
                $('#genderError').text('Please select gender');
                valid= false;
            }

            // Intereset
            if($("input[name='interests']:checked").length === 0){
                $('#interestError').text('Please select at least one interest');
                valid= false;
            }

            // country
            if ($('#country').val() == ''){
                $('#countryError').text('Please select a country!')
                valid= false;
            }

            if(valid){
                alert('Email submitted successfully');
                $('#signupForm')[0].reset();
            }
        });
    });
  </script>
</body>
</html>
```

Signup Form

Name: Sushanka

Email: secret@gmail.com

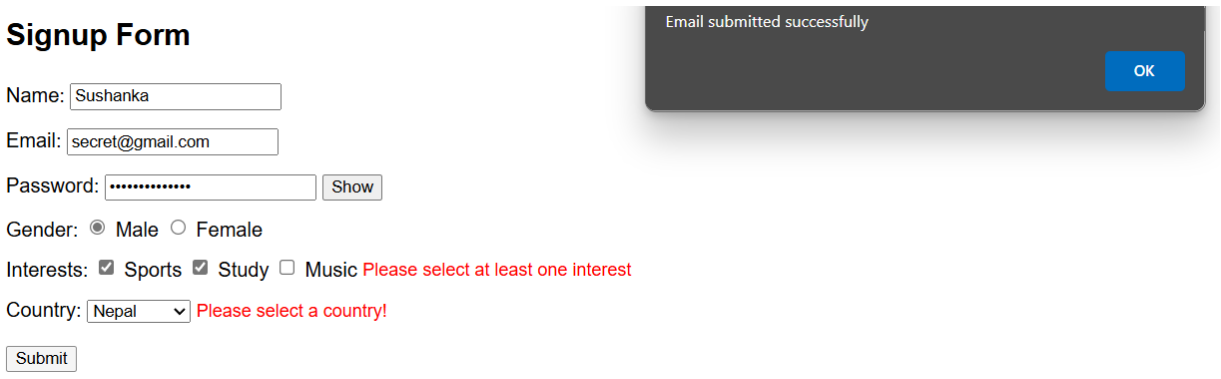
Password: ••••••••••••••• Show

Gender: Male Female

Interests: Sports Study Music Please select at least one interest

Country: ---select--- Please select a country!

Submit



31 c

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery Event</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <style>
        body{
            text-align: center;
        }
        #box {
            width: 200px;
            height: 200px;
            background-color: lightgreen;
            /* text-align: center; */
            line-height: 200px;
            margin: 20px auto;
            font-size: 18px;
            cursor: pointer;
        }
        #output {
            margin-top: 10px;
            /* text-align: center; */
            font-weight: bold;
        }
        form{
            /* text-align: center; */
            margin-top: 30px;
        }

    </style>
</head>
```

```
<body>
    <h3>jQuery Event Handling</h3>
    <div id="box">Hover or click me</div>
    <div id="output"></div>

    <form id="myForm">
        <label>Enter your Name:
            <input type="text" id="name">
            <button type="submit">Submit</button>
        </label>
    </form>

    <script>
        $(function() {
            //click event
            $('#box').click(function () {
                $('#output').text('Box Clicked!')
            })

            // double click event
            $('#box').dblclick(function () {
                $('#output').text('Box Double Clicked!')
            })

            // mouse hover event
            $('#box').mouseenter(function () {
                $(this).css('background-color', 'silver')
                $('#output').text('Mouse entered the box.')
            })

            // mouse leave event
            $('#box').mouseleave(() =>{
                $('#output').text('Mouse leave the box.')
```

```
        })
        // Key dowm event
        $('#name').keydown((e) => {
            $('#output').text('Your pressed: ' + e.key)
        })

        // Form submit event
        $('#myForm').submit(function (e) {
            e.preventDefault()
            let name= $('#name').val()
            $('#output').text('Form Submitted! Hello, ' + name)
        })
    })

  </script>

</body>
</html>
```

jQuery Event Handling

Hover or click me

Your pressed: a

Enter your Name: Sushanka Submit

jQuery Event Handling

Hover or click me

Form Submitted! Hello, Sushanka

Enter your Name: Sushanka Submit

31. d-f

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>jQuery DOM Manipulation</title>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <style>
        div{
            margin: 20px auto;
            width: 400px;
            padding: 10px;
        }
        #content{
            border: 1px solid #333;
            border-radius: 8px;
            background-color: #f2f3f4;
        }
        button{
            margin: 5px;
        }

    </style>
</head>
<body>
    <h2 style="text-align: center;">jQuery DOM Manipulation</h2>
    <div id="content">
        <p id="para">This is a paragraph.</p>
        <img id="image" src="https://www.copahost.com/blog/wp-
content/uploads/2019/07/imgsize2-768x466.png" alt="Placeholder Image"
height="250px" width="400px">
    </div>
```

```
<div>
    <button id="changeText">Change Text</button>
    <button id="changeHTML">Change HTML</button>
    <button id="addElement">Add Element</button>
    <button id="removeElement">Remove Element</button>
    <button id="showImage">Show Image</button>
    <button id="hideImage">Hide Image</button>
    <button id="toggleImage">Toggle Image</button>
    <button id="changeStyle">Change Style</button>
    <button id="slideToggle">Slide Toggle</button>
    <button id="fadeToggle">Fade Toggle</button>
</div>

<script>
    $(document).ready(function() {
        // change text of a paragraph
        $('#changeText').click(() =>{
            $('#para').text('Text chaged using jQuery')
        })

        // change HTML Content
        $('#changeHTML').click(() =>{
            $('#para').html('<strong>This is now bold HTML content!</strong>')
        })

        // add new element
        $('#addElement').click(() =>{
            $('#content').append('<p class="newPara">New paragraph added!</p>')
        })

        // remove paragraph
        $('#removeElement').click(() =>{
            $('.newPara').last().remove()
        })
```

```
        // show image visibility
        $('#showImage').click(() =>{
            $('#image').show()
        })

        // hide image visibility
        $('#hideImage').click(() =>{
            $('#image').hide()
        })

        // toggle image visibility
        $('#toggleImage').click(() =>{
            $('#image').toggle()
        })

        // slide toggle
        $('#slideToggle').click(() =>{
            $('#image').slideToggle()
        })

        // fade toggle
        $('#fadeToggle').click(() =>{
            $('#image').fadeToggle()
        })

        // change style
        $('#changeStyle').click(() =>{
            $('#content').css({
                "background-color": "#dff0d8",
                "color": "#3c763d",
                "border": "2px dashed #3c763d"
            })
        })
    })
</script></body></html>
```

jQuery DOM Manipulation

This is now bold HTML content!

New paragraph added!

Change Text | Change HTML | Add Element | Remove Element | Show Image | Hide Image | Toggle Image | Change Style | Slide Toggle | Fade Toggle

jQuery DOM Manipulation

This is now bold HTML content!

New paragraph added!

Change Text | Change HTML | Add Element | Remove Element | Show Image | Hide Image | Toggle Image | Change Style | Slide Toggle | Fade Toggle

32.a

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <h4> Serialization and Deserialization of JSON (Stringifying and
Parsing)</h4>
   <pre id="preety">show here</pre>
   <script>
      const student_record= {
      "students": [
         {
            "name": "ram",
            "age": 42,
            "grade": "C",
            "subjects": [
               "DAA",
               "WEB"
            ],
            "contact_detail": {
               "email": "ram@gmail.com",
               "phone": 980000000000
            }
         },
         {
            "name": "hari",
            "age": 22,
            "grade": "A",
            "subjects": ["DAA","WEB"],
            "contact_detail": {
               "email": "ram@gmail.com",
```

```
            "phone": 980000000000
        }
    },
    {
        "name": "ram",
        "age": 42,
        "grade": "C",
        "subjects": ["DAA","WEB"],
        "contact_detail": {
            "email": "ram@gmail.com",
            "phone": 980000000000
        }
    }
]
}
// 1. Serialization: Convert JavaScript object to JSON string
const preetyJSON= JSON.stringify(student_record, null, 2)

console.log(preetyJSON)
document.getElementById('preety').innerText= preetyJSON

// 2. Deserialization: Convert JSON string back to JavaScript object
// const deserializedObject = JSON.parse(preetyJSON);

</script>

</body>
</html>
```

Serialization and Deserialization of JSON (Stringifying and Parsing)


```
{
  "students": [
    {
      "name": "ram",
      "age": 42,
      "grade": "C",
      "subjects": [
        "DAA",
        "WEB"
      ],
      "contact_detail": {
        "email": "ram@gmail.com",
        "phone": 980000000000
      }
    },
    {
      "name": "hari",
      "age": 22,
      "grade": "A",
      "subjects": [
        "DAA",
        "WEB"
      ],
      "contact_detail": {
        "email": "ram@gmail.com",
        "phone": 980000000000
      }
    },
    {
      "name": "ram",
      "age": 42,
      "grade": "C",
      "subjects": [
        "DAA",
        "WEB"
      ],
      "contact_detail": {
        "email": "ram@gmail.com",
        "phone": 980000000000
      }
    }
```

32.b

```
const fs = require('fs');

// ---------- Read from JSON file ----------
const jsonData = fs.readFileSync('32.b.json', 'utf-8');
const data = JSON.parse(jsonData);

console.log("Original Data:", data);

// ---------- Modify or Add Data ----------
data.grade = 'A+';
data.subjects = ['DAA', 'WEB', 'AI'];

// ---------- Write to JSON file ----------
fs.writeFileSync('32.b.json', JSON.stringify(data, null, 2));

console.log("Data updated and written to 32.b.json!");
```

```
PS C:\Users\me\Desktop\lab\lab 3> node 32.b.js
Original Data: { name: 'Alice', age: 25 }
Data updated and written to 32.b.json!
PS C:\Users\me\Desktop\lab\lab 3>
```

lab 3 > {} 32.b.json > ...

```
{
  "name": "Alice",
  "age": 25,
  "grade": "A+",
  "subjects": [
    "DAA",
    "WEB",
    "AI"
  ]
}
```