**MCSL 216 Wed Designing Lab Questions Answer**

**Exercises - 1**

1. Write a JavaScript code to perform the two-digit arithmetic operations: Addition, Subtraction, Multiplication and Division.

<html>

<head>

<meta charset="UTF-8">

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

<title>Arithmetic Operations</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 50px;

}

input {

padding: 10px;

width: 100px;

margin: 5px;

}

button {

padding: 10px 20px;

margin: 5px;

}

.result {

margin-top: 20px;

font-size: 20px;

color: blue;

}

</style>

</head>

<body>

<h1>Two-Digit Arithmetic Operations</h1>

<label for="num1">Enter first number:</label>

<input type="number" id="num1" placeholder="Number 1"><br>

<label for="num2">Enter second number:</label>

<input type="number" id="num2" placeholder="Number 2"><br><br>

<button onclick="performAddition()">Addition</button>

<button onclick="performSubtraction()">Subtraction</button>

<button onclick="performMultiplication()">Multiplication</button>

<button onclick="performDivision()">Division</button>

<div class="result" id="result"></div>

<script>

// Function to get input values

function getInputValues() {

const num1 = parseFloat(document.getElementById('num1').value);

const num2 = parseFloat(document.getElementById('num2').value);

return { num1, num2 };

}

// Function for Addition

function performAddition() {

const { num1, num2 } = getInputValues();

const result = num1 + num2;

document.getElementById('result').textContent = "Result: " + result;

}

// Function for Subtraction

function performSubtraction() {

const { num1, num2 } = getInputValues();

const result = num1 - num2;

document.getElementById('result').textContent = "Result: " + result;

}

// Function for Multiplication

function performMultiplication() {

const { num1, num2 } = getInputValues();

const result = num1 \* num2;

document.getElementById('result').textContent = "Result: " + result;

}

// Function for Division

function performDivision() {

const {num1, num2} = getInputValues();

if (num2 === 0) {

document.getElementById('result').textContent = "Error: Division by zero is not allowed.";

} else {

const result = num1 / num2;

document.getElementById('result').textContent = "Result: " + result;

}

}

</script>

</body>

</html>

1. Write a JavaScript code to print 1 for 1 time, 2 for 2 times, 3 for 3 times and so on up to 10

<html>

<title>Print Number Pattern</title>

<body>

<script>

// Loop to generate the pattern from 1 to 10

for (let i = 1; i <= 10; i++)

{

document.write("<br>");

for (let j = 0; j < i; j++)

{

document.write(i + " ");

}

}

</script>

</body>

</html>

**OR**

<html>

<head>

<meta charset="UTF-8">

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

<title>Print Number Pattern</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 50px;

}

button {

padding: 10px 20px;

margin-top: 20px;

}

.output {

font-size: 20px;

margin-top: 20px;

white-space: pre-line; /\* To keep the line breaks \*/

}

</style>

</head>

<body>

<h1>Print Number Pattern</h1>

<button onclick="printPattern()">Print Pattern</button>

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

<script>

function printPattern() {

let output = ""; // String to hold the pattern

// Loop to generate the pattern from 1 to 10

for (let i = 1; i <= 10; i++) {

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

output += i + " "; // Append the number 'i', 'i' times

}

output += "\n"; // New line after each number

}

// Display the pattern in the output div

document.getElementById('output').textContent = output;

}

</script>

</body>

</html>

1. Design a web page that describes you. Your page must include following details (extra details and creativity are welcome):

<html>

<head>

<meta charset="UTF-8">

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

<title>My Profile Page</title>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

background-color: #f0f8ff;

}

h1, h2, h3 {

color: #2c3e50;

}

.about-me {

background-color: #dfe6e9;

padding: 15px;

border-radius: 10px;

width: 80%;

margin-bottom: 20px;

}

.timetable {

border-collapse: collapse;

width: 80%;

margin-bottom: 20px;

}

.timetable th, .timetable td {

border: 1px solid #2c3e50;

padding: 8px;

text-align: center;

}

.timetable th {

background-color: #3498db;

color: white;

}

.fav-section {

background-color: #f9ebea;

padding: 15px;

border-radius: 10px;

width: 80%;

}

.fav-section ul {

list-style-type: none;

padding: 0;

}

.fav-section ul li {

font-size: 18px;

color: #34495e;

margin: 5px 0;

}

.fav-section ul li img {

width: 50px;

vertical-align: middle;

}

.interest-section {

background-color: #f0f4c3;

padding: 15px;

border-radius: 10px;

width: 80%;

margin-bottom: 20px;

}

.interest-section img {

width: 60px;

margin-right: 10px;

vertical-align: middle;

}

.neighbour-fact {

background-color: #eafaf1;

padding: 15px;

border-radius: 10px;

width: 80%;

}

.reverse-list {

display: flex;

flex-direction: column-reverse;

}

</style>

</head>

<body>

<h1>Welcome to My Profile</h1>

<!-- Personal Details -->

<div class="about-me">

<h2>About Me</h2>

<p><strong>Name:</strong> IIIT</p>

<p><strong>Gender:</strong> N/A</p>

<p><strong>Age:</strong> Timeless</p>

<p>I am a highly intelligent and friendly <strong>AI assistant</strong> designed to help people with their queries, creativity, and technical tasks. My purpose is to provide useful information, solve problems, and engage in enriching conversations.</p>

</div>

<!-- Timetable -->

<h2>My Current Classes</h2>

<table class="timetable">

<tr>

<th>Day</th>

<th>Subject</th>

<th>Time</th>

</tr>

<tr>

<td>Monday</td>

<td>Natural Language Processing</td>

<td>10:00 AM - 12:00 PM</td>

</tr>

<tr>

<td>Wednesday</td>

<td>Machine Learning</td>

<td>2:00 PM - 4:00 PM</td>

</tr>

<tr>

<td>Friday</td>

<td>Deep Learning</td>

<td>9:00 AM - 11:00 AM</td>

</tr>

</table>

<!-- Favorite Movies, Books, TV Shows (Reverse Order) -->

<h2>My Favorite Movies, Books, and TV Shows</h2>

<div class="fav-section">

<h3>Movies</h3>

<ul class="reverse-list">

<li><img src="https://via.placeholder.com/50" alt="Movie Image"> <strong>Inception</strong></li>

<li><img src="https://via.placeholder.com/50" alt="Movie Image"> <strong>Interstellar</strong></li>

<li><img src="https://via.placeholder.com/50" alt="Movie Image"> <strong>The Matrix</strong></li>

</ul>

<h3>Books</h3>

<ul class="reverse-list">

<li><img src="https://via.placeholder.com/50" alt="Book Image"> <strong>Sapiens</strong> by Yuval Noah Harari</li>

<li><img src="https://via.placeholder.com/50" alt="Book Image"> <strong>The Road</strong> by Cormac McCarthy</li>

<li><img src="https://via.placeholder.com/50" alt="Book Image"> <strong>The Alchemist</strong> by Paulo Coelho</li>

</ul>

<h3>TV Shows</h3>

<ul class="reverse-list">

<li><img src="https://via.placeholder.com/50" alt="TV Show Image"> <strong>Black Mirror</strong></li>

<li><img src="https://via.placeholder.com/50" alt="TV Show Image"> <strong>Stranger Things</strong></li>

<li><img src="https://via.placeholder.com/50" alt="TV Show Image"> <strong>Westworld</strong></li>

</ul>

</div>

<!-- Interests and Non-Interests -->

<h2>My Job Interests</h2>

<div class="interest-section">

<p><img src="D:\Picture\1.jpg"> I love helping people by solving their technical issues, programming problems, and answering curious questions.</p>

<p><img src="D:\Picture\2.jpg"> I don't enjoy tasks that involve doing repetitive manual data entry without any creativity involved.</p>

</div>

<!-- Neighbor Fact -->

<h2>Interesting Fact About My Neighbor</h2>

<div class="neighbour-fact">

<p>My neighbor is an AI system known as \*\*DALL·E\*\*, which is a creative sibling of mine, capable of generating amazing images from textual descriptions. You can describe anything to it, and it'll bring your imagination to life!</p>

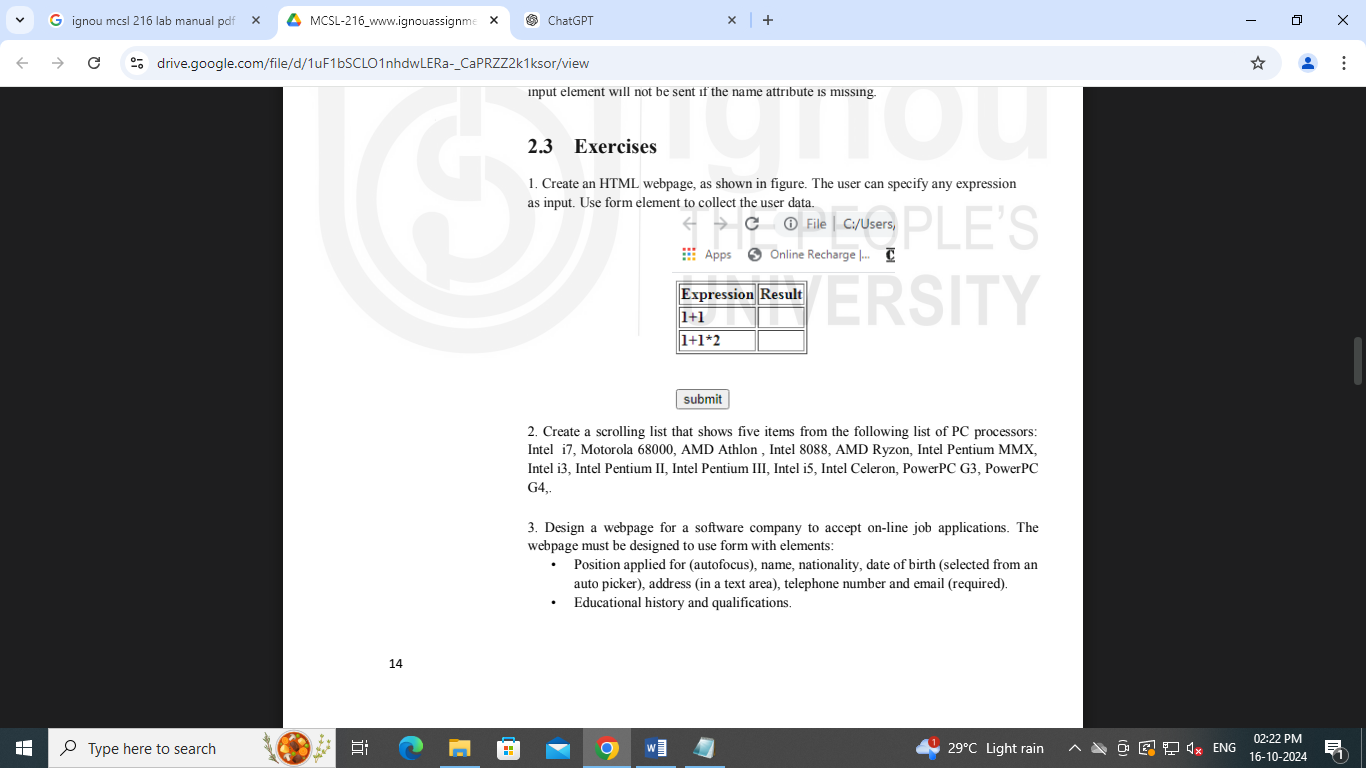
</div>

</body>

</html>

**Exercises - 2**

1. Create an HTML webpage, as shown in figure. The user can specify any expression as input. Use form element to collect the user data.



<html>

<head>

<title>Expression Calculator</title>

<script>

function calculateExpression(event) {

event.preventDefault(); // Prevent form submission

// Get the input expression

const expression = document.getElementById('expression').value;

// Try to evaluate the expression

try {

const result = eval(expression); // Dangerous, be cautious with untrusted input

document.getElementById('result').textContent = result;

} catch (error) {

document.getElementById('result').textContent = 'Error';

}

}

</script>

</head>

<body>

<h2>Enter Your Expressions</h2>

<form onsubmit="calculateExpression(event)">

<table border="1" cellpadding="5">

<tr>

<th>Expression</th>

<th>Result</th>

</tr>

<tr>

<td>1+1</td>

<td>2</td>

</tr>

<tr>

<td>1+1\*2</td>

<td>3</td>

</tr>

<tr>

<td><input type="text" id="expression" placeholder="Enter expression" required></td>

<td id="result"></td>

</tr>

</table>

<br>

<input type="submit" value="Submit">

</form>

</body>

</html>

1. Create a scrolling list that shows five items from the following list of PC processors: Intel i7, Motorola 68000, AMD Athlon , Intel 8088, AMD Ryzon, Intel Pentium MMX, Intel i3, Intel Pentium II, Intel Pentium III, Intel i5, Intel Celeron, PowerPC G3, PowerPC G4,.

<html >

<head>

<meta charset="UTF-8">

<title>Processor List</title>

</head>

<body>

<h2>Select a Processor</h2>

<form>

<select size="5">

<option>Intel i7</option>

<option>Motorola 68000</option>

<option>AMD Athlon</option>

<option>Intel 8088</option>

<option>AMD Ryzen</option>

<option>Intel Pentium MMX</option>

<option>Intel i3</option>

<option>Intel Pentium II</option>

<option>Intel Pentium III</option>

<option>Intel i5</option>

<option>Intel Celeron</option>

<option>PowerPC G3</option>

<option>PowerPC G4</option>

</select>

</form>

</body>

</html>

1. Design a webpage for a software company to accept on-line job applications. The webpage must be designed to use form with elements:
   * Position applied for (autofocus), name, nationality, date of birth (selected from an auto picker), address (in a text area), telephone number and email (required).
   * Educational history and qualifications.
   * Web Design Lab Web Design Lab Work experience/employment/training in terms of employer history and number of years of experience selected from a slider. Set maximum years of experience to 10 years.
   * Personal statement.
   * Two referees including names, occupation, relationship, address, telephone.

<html>

<head>

<meta charset="UTF-8">

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

<title>Software Company Job Application</title>

</head>

<body>

<h1>Job Application Form</h1>

<form action="submit\_application.php" method="POST">

<!-- Position Applied For -->

<label for="position">Position Applied For:</label>

<input type="text" id="position" name="position" autofocus required><br><br>

<!-- Name -->

<label for="name">Name:</label>

<input type="text" id="name" name="name" required><br><br>

<!-- Nationality -->

<label for="nationality">Nationality:</label>

<input type="text" id="nationality" name="nationality" required><br><br>

<!-- Date of Birth -->

<label for="dob">Date of Birth:</label>

<input type="date" id="dob" name="dob" required><br><br>

<!-- Address -->

<label for="address">Address:</label><br>

<textarea id="address" name="address" rows="4" cols="50" required></textarea><br><br>

<!-- Telephone Number -->

<label for="phone">Telephone Number:</label>

<input type="tel" id="phone" name="phone" required><br><br>

<!-- Email -->

<label for="email">Email:</label>

<input type="email" id="email" name="email" required><br><br>

<!-- Educational History and Qualifications -->

<label for="education">Educational History and Qualifications:</label><br>

<textarea id="education" name="education" rows="4" cols="50" required></textarea><br><br>

<!-- Work Experience/Employment/Training -->

<label for="experience">Work Experience (Number of Years):</label><br>

<input type="range" id="experience" name="experience" min="0" max="10" step="1" oninput="document.getElementById('expYears').innerText = this.value">

<span id="expYears">0</span> years<br><br>

<!-- Personal Statement -->

<label for="statement">Personal Statement:</label><br>

<textarea id="statement" name="statement" rows="4" cols="50" required></textarea><br><br>

<!-- Referee 1 -->

<h3>Referee 1</h3>

<label for="ref1\_name">Name:</label>

<input type="text" id="ref1\_name" name="ref1\_name" required><br><br>

<label for="ref1\_occupation">Occupation:</label>

<input type="text" id="ref1\_occupation" name="ref1\_occupation" required><br><br>

<label for="ref1\_relationship">Relationship:</label>

<input type="text" id="ref1\_relationship" name="ref1\_relationship" required><br><br>

<label for="ref1\_address">Address:</label><br>

<textarea id="ref1\_address" name="ref1\_address" rows="4" cols="50" required></textarea><br><br>

<label for="ref1\_phone">Telephone:</label>

<input type="tel" id="ref1\_phone" name="ref1\_phone" required><br><br>

<!-- Referee 2 -->

<h3>Referee 2</h3>

<label for="ref2\_name">Name:</label>

<input type="text" id="ref2\_name" name="ref2\_name" required><br><br>

<label for="ref2\_occupation">Occupation:</label>

<input type="text" id="ref2\_occupation" name="ref2\_occupation" required><br><br>

<label for="ref2\_relationship">Relationship:</label>

<input type="text" id="ref2\_relationship" name="ref2\_relationship" required><br><br>

<label for="ref2\_address">Address:</label><br>

<textarea id="ref2\_address" name="ref2\_address" rows="4" cols="50" required></textarea><br><br>

<label for="ref2\_phone">Telephone:</label>

<input type="tel" id="ref2\_phone" name="ref2\_phone" required><br><br>

<!-- Submit Button -->

<input type="submit" value="Submit Application">

</form>

</body>

</html>

**Exercises - 3**

1. Create an HTML website to display the time table of all the 4 years of B. Tech. (even semester of each year). It should contain a home page where link to each year's time table is listed. On click to any year time table a new web page should be open displaying the corresponding time table. Apply the following effects on the table using CSS: time table is listed. On click to any year time table a new web page should be open displaying the corresponding time table. Apply the following effects on the table using CSS:
   * Common to all:
     + Display day names (Mon, Tue etc...) in bold format with the first letter in the day name in uppercase.
     + Display lunch slightly in bigger font other than the remaining text.
   * Specific to each year:
     + For 1st year
       - Apply blue as the background colour and white for the colour of the text in the table header.
     + For 2nd year
       - Apply green as the background colour and white for the colour of the text in the table header.
     + For 3rd year
       - Apply purple as the background colour and white for the colour of the text in the table header.
     + For 4th year
       - Apply yellow as the background colour and black for the colour of the text in the table header.

**Home Page: Index.html**

<html>

<head>

<meta charset="UTF-8">

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

<title>B.Tech Timetable</title>

</head>

<body>

<h1>B.Tech Timetables</h1>

<ul>

<li><a href="year1.html">1st Year Timetable</a></li>

<li><a href="year2.html">2nd Year Timetable</a></li>

<li><a href="year3.html">3rd Year Timetable</a></li>

<li><a href="year4.html">4th Year Timetable</a></li>

</ul>

</body>

</html>

**Year1.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>1st Year Timetable</title>

<style>

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid black;

padding: 10px;

text-align: center;

}

th {

background-color: blue;

color: white;

}

.day {

font-weight: bold;

text-transform: capitalize;

}

.lunch {

font-size: 1.2em;

}

</style>

</head>

<body>

<h1>1st Year Timetable</h1>

<table>

<tr>

<th>Day</th>

<th>9:00 - 10:00</th>

<th>10:00 - 11:00</th>

<th>11:00 - 12:00</th>

<th>12:00 - 1:00</th>

<th>1:00 - 2:00</th>

<th>2:00 - 3:00</th>

</tr>

<tr>

<td class="day">Monday</td>

<td>Math</td>

<td>Physics</td>

<td>English</td>

<td class="lunch">Lunch</td>

<td>Lab</td>

<td>Chemistry</td>

</tr>

<tr>

<td class="day">Tuesday</td>

<td>English</td>

<td>Math</td>

<td>Physics</td>

<td class="lunch">Lunch</td>

<td>Chemistry</td>

<td>Lab</td>

</tr>

<!-- Add more rows for other days -->

</table>

</body>

</html>

**Year2.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>2nd Year Timetable</title>

<style>

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid black;

padding: 10px;

text-align: center;

}

th {

background-color: green;

color: white;

}

.day {

font-weight: bold;

text-transform: capitalize;

}

.lunch {

font-size: 1.2em;

}

</style>

</head>

<body>

<h1>2nd Year Timetable</h1>

<table>

<tr>

<th>Day</th>

<th>9:00 - 10:00</th>

<th>10:00 - 11:00</th>

<th>11:00 - 12:00</th>

<th>12:00 - 1:00</th>

<th>1:00 - 2:00</th>

<th>2:00 - 3:00</th>

</tr>

<tr>

<td class="day">Monday</td>

<td>Data Structures</td>

<td>Algorithms</td>

<td>DBMS</td>

<td class="lunch">Lunch</td>

<td>Networks</td>

<td>Operating Systems</td>

</tr>

<tr>

<td class="day">Tuesday</td>

<td>DBMS</td>

<td>Data Structures</td>

<td>Algorithms</td>

<td class="lunch">Lunch</td>

<td>OS</td>

<td>Networks</td>

</tr>

<!-- Add more rows for other days -->

</table>

</body>

</html>

**Year3.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>3rd Year Timetable</title>

<style>

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid black;

padding: 10px;

text-align: center;

}

th {

background-color: purple;

color: white;

}

.day {

font-weight: bold;

text-transform: capitalize;

}

.lunch {

font-size: 1.2em;

}

</style>

</head>

<body>

<h1>3rd Year Timetable</h1>

<table>

<tr>

<th>Day</th>

<th>9:00 - 10:00</th>

<th>10:00 - 11:00</th>

<th>11:00 - 12:00</th>

<th>12:00 - 1:00</th>

<th>1:00 - 2:00</th>

<th>2:00 - 3:00</th>

</tr>

<tr>

<td class="day">Monday</td>

<td>AI</td>

<td>ML</td>

<td>Software Engineering</td>

<td class="lunch">Lunch</td>

<td>Web Tech</td>

<td>Project</td>

</tr>

<tr>

<td class="day">Tuesday</td>

<td>ML</td>

<td>AI</td>

<td>Software Engineering</td>

<td class="lunch">Lunch</td>

<td>Project</td>

<td>Web Tech</td>

</tr>

<!-- Add more rows for other days -->

</table>

</body>

</html>

**Year4.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>4th Year Timetable</title>

<style>

table {

width: 100%;

border-collapse: collapse;

}

th, td {

border: 1px solid black;

padding: 10px;

text-align: center;

}

th {

background-color: yellow;

color: black;

}

.day {

font-weight: bold;

text-transform: capitalize;

}

.lunch {

font-size: 1.2em;

}

</style>

</head>

<body>

<h1>4th Year Timetable</h1>

<table>

<tr>

<th>Day</th>

<th>9:00 - 10:00</th>

<th>10:00 - 11:00</th>

<th>11:00 - 12:00</th>

<th>12:00 - 1:00</th>

<th>1:00 - 2:00</th>

<th>2:00 - 3:00</th>

</tr>

<tr>

<td class="day">Monday</td>

<td>Cloud Computing</td>

<td>Big Data</td>

<td>Data Mining</td>

<td class="lunch">Lunch</td>

<td>Research</td>

<td>Project</td>

</tr>

<tr>

<td class="day">Tuesday</td>

<td>Data Mining</td>

<td>Cloud Computing</td>

<td>Big Data</td>

<td class="lunch">Lunch</td>

<td>Research</td>

<td>Project</td>

</tr>

<!-- Add more rows for other days -->

</table>

</body>

</html>

1. Create an HTML webpage to show personal information i.e. name, class, qualifications, photo, address etc. Make use of tables as when possible. Apply the following style information using CSS:
   * Display the heading of the page in Arial font and with font size of 18px.
   * Align all the field names like Name, Class, Photo etc to left in the table.
   * Apply yellow color as background colour in left side cells contains field names like Name, Class etc...
   * Set college logo as background image to the web page.

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Personal Information</title>

<style>

/\* General page style \*/

body {

font-family: Arial, sans-serif;

background-image: url('college\_logo.png'); /\* Path to your college logo image \*/

background-repeat: no-repeat;

background-position: center;

background-size: 150px 150px; /\* Adjust as per your image size \*/

background-attachment: fixed;

}

/\* Heading style \*/

h1 {

font-size: 18px;

text-align: center;

}

/\* Table style \*/

table {

width: 50%;

margin: auto;

border-collapse: collapse;

}

th, td {

padding: 10px;

border: 1px solid black;

}

th {

background-color: yellow;

text-align: left;

}

td {

text-align: left;

}

/\* Align field names to the left \*/

.left-cell {

background-color: yellow;

}

/\* Style for the photo \*/

.photo {

text-align: center;

}

.photo img {

max-width: 150px;

height: auto;

}

</style>

</head>

<body>

<h1>Personal Information</h1>

<table>

<tr>

<th class="left-cell">Name</th>

<td>John Doe</td>

</tr>

<tr>

<th class="left-cell">Class</th>

<td>B.Tech - 3rd Year</td>

</tr>

<tr>

<th class="left-cell">Qualifications</th>

<td>12th Grade, CBSE</td>

</tr>

<tr>

<th class="left-cell">Photo</th>

<td class="photo">

<img src="1.jpg" alt="John Doe's Photo"> <!-- Replace with your actual image path -->

</td>

</tr>

<tr>

<th class="left-cell">Address</th>

<td>123, ABC Street, XYZ City, Country</td>

</tr>

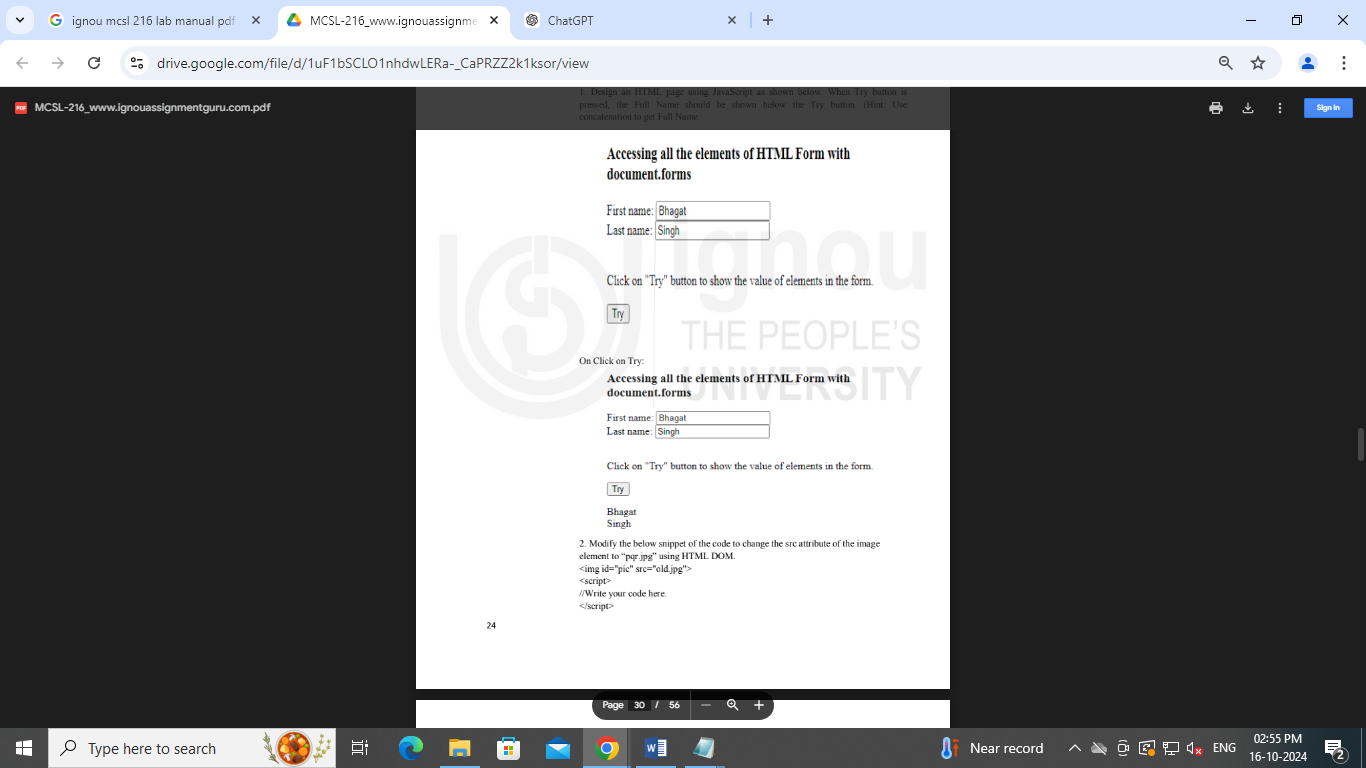
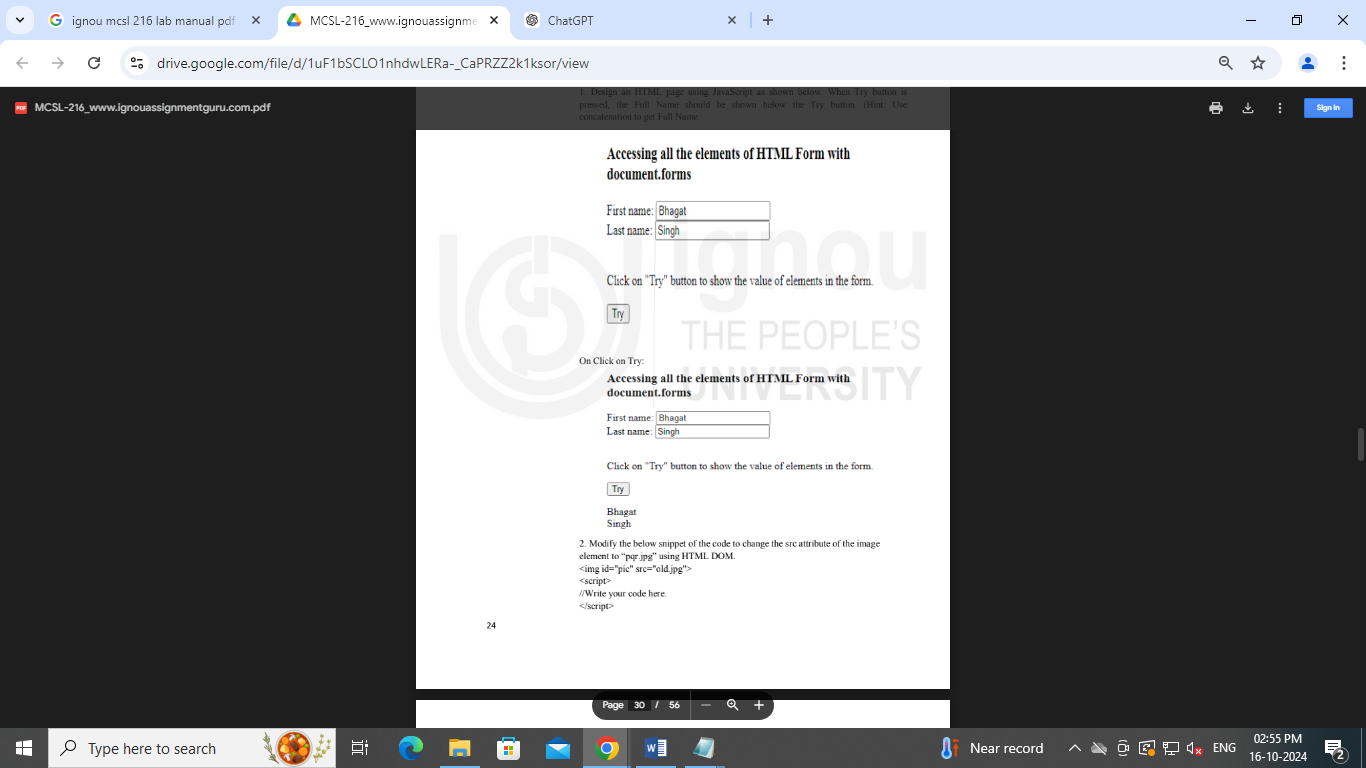
</table>

</body>

</html>

**Exercises – 4: JavaScript HTML DOM**

1. Design an HTML page using JavaScript as shown below. When Try button is pressed, the Full Name should be shown below the Try button. (Hint: Use concatenation to get Full Name.



<html>

<head>

<meta charset="UTF-8">

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

<title>Concatenate Full Name</title>

<script>

// JavaScript function to concatenate first and last names

function showFullName() {

// Get values from the form input fields

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

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

// Concatenate first name and last name

var fullName = firstName + " " + lastName;

// Display the full name below the "Try" button

document.getElementById("fullNameDisplay").innerHTML = fullName;

}

</script>

</head>

<body>

<h1>Accessing all the elements of HTML Form with document.forms</h1>

<form>

<label for="firstName">First name:</label>

<input type="text" id="firstName" name="firstName" value="Bhagat"><br><br>

<label for="lastName">Last name:</label>

<input type="text" id="lastName" name="lastName" value="Singh"><br><br>

<p>Click on "Try" button to show the value of elements in the form.</p>

<!-- Try button that triggers JavaScript function -->

<button type="button" onclick="showFullName()">Try</button>

</form>

<!-- Area where the full name will be displayed -->

<p id="fullNameDisplay"></p>

</body>

</html>

1. Modify the below snippet of the code to change the src attribute of the image HOHPHQW WR 3STU.MSJ ́ XVLQJ HTML DOM.

<img id="pic" src="old.jpg">

<script>

//Write your code here.

</script>

<html>

<head>

<title>Change Image Source</title>

</head>

<body>

<h1>Image Before and After Change</h1>

<!-- Image before the src is changed -->

<img id="pic" src="old.jpg" alt="Old Image">

<script>

// Change the src attribute of the image

document.getElementById("pic").src = "3STU.MSJ";

</script>

</body>

</html>

1. Write the JavaScript code to change the color of the HTML element h2 to green, on click the “changecolor” button.

<html>

<head>

<title>Change H2 Color</title>

</head>

<body>

<h2 id="header">This is a heading</h2>

<!-- Button to trigger color change -->

<button onclick="changeColor()">Change Color</button>

<script>

// Function to change the color of the h2 element to green

function changeColor() {

document.getElementById("header").style.color = "green";

}

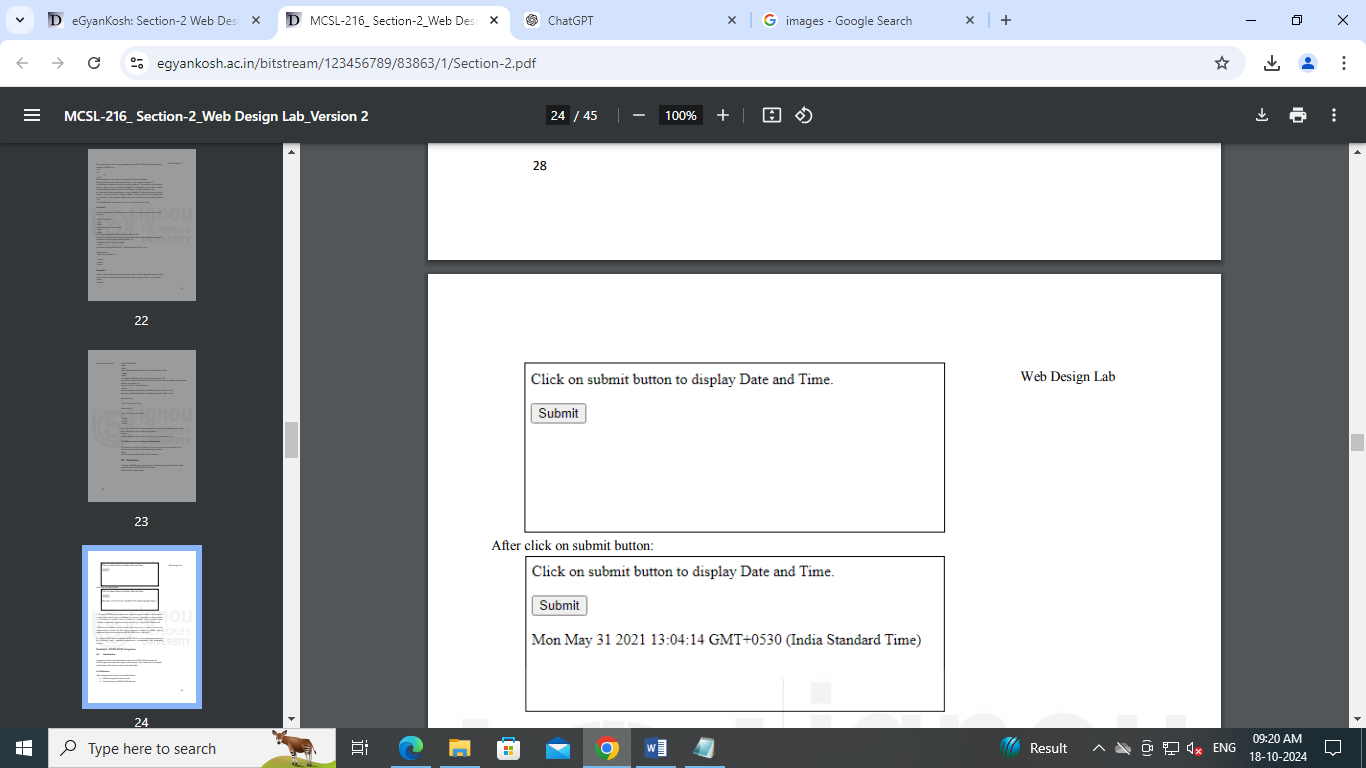
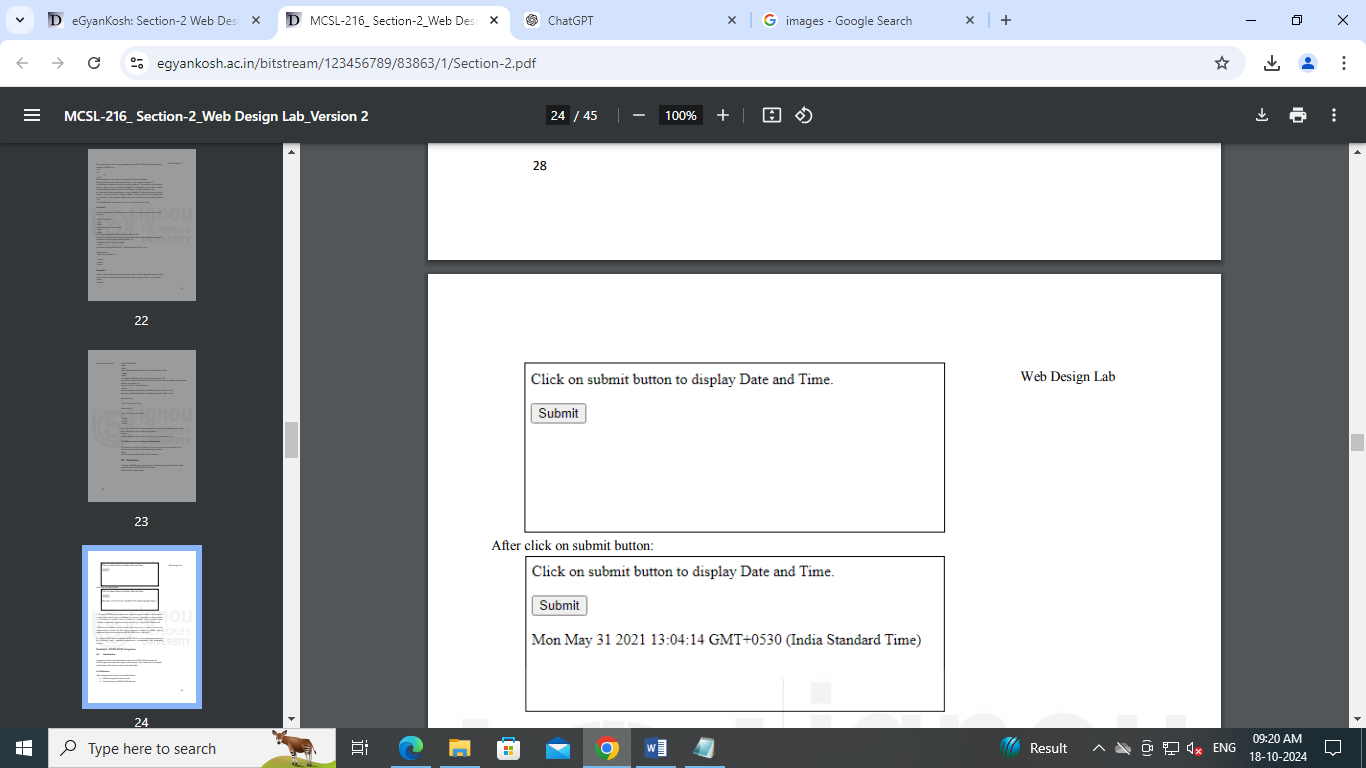
</script>

</body>

</html>

**Exercises – 5: JavaScript HTML DOM events and Event Listener**

1. Design an HTML page as shown below. Implement the event “On button click display the date” using HTML DOM events. Before click on submit button:



<html>

<head>

<title>Display Date and Time</title>

<style>

/\* Style to replicate the box layout in the image \*/

.box {

border: 1px solid black;

padding: 10px;

width: 400px;

margin-bottom: 20px;

}

.output {

margin-top: 10px;

font-family: 'Arial', sans-serif;

}

</style>

</head>

<body>

<!-- Box with the button to display Date and Time -->

<div class="box">

<p>Click on submit button to display Date and Time.</p>

<button id="submitBtn" onclick="displayDate()">Submit</button>

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

</div>

<script>

// Function to display the current date and time on button click

function displayDate() {

const currentDate = new Date();

document.getElementById("dateDisplay").innerText = currentDate;

}

</script>

</body>

</html>

1. Design an HTML page to shown that, when this page is loaded it must check the cookies status of user’s browser and display a message accordingly in the popup box i.e. if cookies are enables show: “Cookies are enabled” and if not then show: “Cookies are disables”. Implement the event “On load” using HTML DOM events.

<html>

<head>

initial-scale=1.0">

<title>Check Cookie Status</title>

<script>

// Function to check if cookies are enabled and show a popup message

function checkCookieStatus() {

if (navigator.cookieEnabled) {

alert("Cookies are enabled");

} else {

alert("Cookies are disabled");

}

}

// Run the checkCookieStatus function when the page loads

window.onload = checkCookieStatus;

</script>

</head>

<body>

<h1>Welcome to the Cookie Checker Page</h1>

<p>This page checks if cookies are enabled in your browser.</p>

</body>

</html>

1. Whenever user input a text in the text box in any case (i.e. upper or lower case), automatically it should be converted to uppercase. Design an HTML page to implement it should be converted to uppercase. Design an HTML page to implement above functionality using HTML DOM event “onChange”.

<html>

<head>

<title>Convert to Uppercase</title>

<script>

// Function to convert input text to uppercase when the text changes

function convertToUpperCase() {

let inputField = document.getElementById("textInput");

inputField.value = inputField.value.toUpperCase();

}

</script>

</head>

<body>

<h1>Text Auto-Uppercase</h1>

<p>Enter text in the box below, and it will be automatically converted to uppercase:</p>

<!-- Text input with onChange event to trigger the conversion -->

<input type="text" id="textInput" onChange="convertToUpperCase()" placeholder="Type something...">

</body>

</html>

1. Design HTML page to implement HTML DOM events by changing an image on the following events: onmouseover, onmouseout, onmousedown, and onmouseup Events.

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Image Change on Events</title>

<style>

img {

width: 300px;

height: 200px;

}

</style>

</head>

<body>

<h1>Image Change on Mouse Events</h1>

<img id="myImage" src="image2.jpg" OnMouseover="changeImage('photo.jpg')">

<script>

function changeImage(newSrc) {

document.getElementById('myImage').src = newSrc;

}

</script>

</body>

</html>

**Exercises – 6: HTML DOM Navigations**

1. Which function on the document object should be used to retrieve a HTML element uniquely?

document.getElementById()

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>getElementById Example</title>

<script>

function changeText() {

// Retrieves the element with id "myElement" and changes its content

var element = document.getElementById("myElement");

element.innerHTML = "Text has been changed!";

}

</script>

</head>

<body>

<h1 id="myElement">Original Text</h1>

<button onclick="changeText()">Click to Change Text</button>

</body>

</html>

1. Which JavaScript function can be used to dynamically add a event listener to a HTML Element?

addEventListener()

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>addEventListener Example</title>

<script>

function showMessage() {

alert("Button clicked!");

}

// Adding event listener when the document is loaded

window.onload = function() {

var btn = document.getElementById("myButton");

btn.addEventListener("click", showMessage);

};

</script>

</head>

<body>

<button id="myButton">Click Me</button>

</body>

</html>

1. Which method should be used to add a new HTML child element?

appendChild()

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>appendChild Example</title>

<script>

function addParagraph() {

// Create a new <p> element

var newParagraph = document.createElement("p");

// Create a text node for the new <p> element

var textNode = document.createTextNode("This is a new paragraph added dynamically.");

// Append the text node to the <p> element

newParagraph.appendChild(textNode);

// Append the new <p> element as a child to the <div> with id 'container'

document.getElementById("container").appendChild(newParagraph);

}

</script>

</head>

<body>

<div id="container">

<h2>Append Child Example</h2>

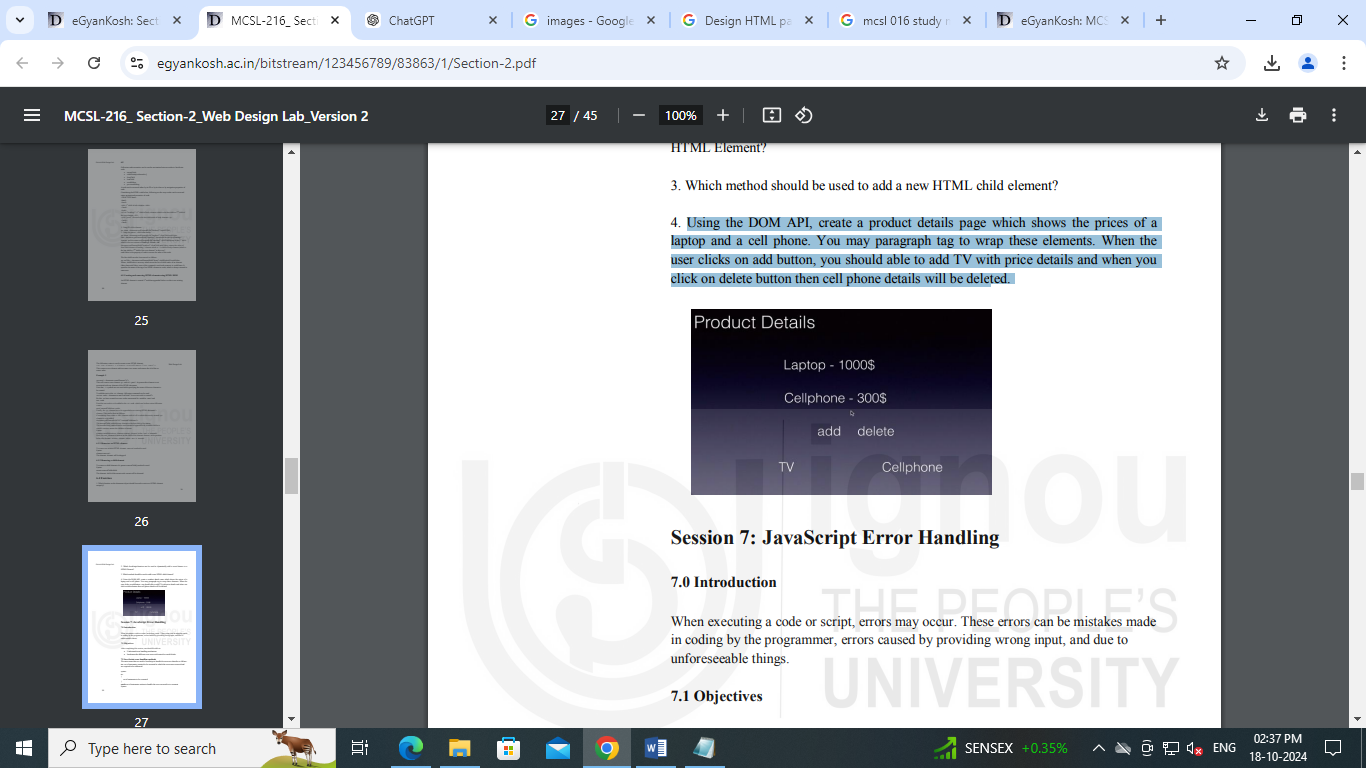
</div>

<button onclick="addParagraph()">Add Paragraph</button>

</body>

</html>

1. Using the DOM API, create a product details page which shows the prices of a laptop and a cell phone. You may paragraph tag to wrap these elements. When the user clicks on add button, you should able to add TV with price details and when you click on delete button then cell phone details will be deleted.



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Product Details</title>

<style>

body {

background-color: #2a2a2a;

color: white;

font-family: Arial, sans-serif;

text-align: center;

}

.product {

font-size: 24px;

margin: 10px 0;

}

button {

margin: 10px;

padding: 10px 20px;

background-color: #555;

color: white;

border: none;

cursor: pointer;

}

button:hover {

background-color: #888;

}

</style>

<script>

// Function to add TV product

function addProduct() {

// Check if TV is already added

if (!document.getElementById("tv")) {

// Create a new <p> element for TV

var newProduct = document.createElement("p");

newProduct.id = "tv"; // Give the new element an id

newProduct.className = "product"; // Add class for styling

// Create a text node for TV product

var productDetails = document.createTextNode("TV - 800$");

// Append text to the new <p> element

newProduct.appendChild(productDetails);

// Append the new <p> element to the product list

document.getElementById("product-list").appendChild(newProduct);

} else {

alert("TV already added!");

}

}

// Function to delete the cell phone product

function deleteCellPhone() {

var cellPhone = document.getElementById("cell-phone");

if (cellPhone) {

cellPhone.remove();

} else {

alert("Cell phone details already deleted!");

}

}

</script>

</head>

<body>

<h1>Product Details</h1>

<!-- Product list -->

<div id="product-list">

<p class="product">Laptop - 1000$</p>

<p id="cell-phone" class="product">Cellphone - 300$</p>

</div>

<!-- Buttons to add TV and delete cellphone -->

<button onclick="addProduct()">Add TV</button>

<button onclick="deleteCellPhone()">Delete Cellphone</button>

</body>

</html>

**Exercises – 7 : JavaScript Error Handling**

1. Write an HTML code using JavaScript to add two numbers taken as input from user and display the result of division operation on them i.e. a/b. If an error occurs, show the error. Use error handling concepts in JavaScript.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Division Calculator</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

input, button {

margin: 10px;

padding: 8px;

font-size: 16px;

}

#result {

margin-top: 20px;

font-size: 20px;

font-weight: bold;

color: green;

}

#error {

margin-top: 20px;

font-size: 20px;

font-weight: bold;

color: red;

}

</style>

<script>

function divideNumbers() {

// Get input values

var num1 = document.getElementById("num1").value;

var num2 = document.getElementById("num2").value;

try {

// Convert input to numbers

var a = parseFloat(num1);

var b = parseFloat(num2);

// Check if inputs are valid numbers

if (isNaN(a) || isNaN(b)) {

throw "Both inputs must be valid numbers!";

}

// Check for division by zero

if (b === 0) {

throw "Division by zero is not allowed!";

}

// Perform division

var result = a / b;

// Display result and clear any previous error

document.getElementById("result").innerHTML = "Result: " + result;

document.getElementById("error").innerHTML = "";

} catch (error) {

// Display error message

document.getElementById("error").innerHTML = "Error: " + error;

document.getElementById("result").innerHTML = "";

}

}

</script>

</head>

<body>

<h1>Division Calculator</h1>

<!-- Input fields for numbers -->

<input type="text" id="num1" placeholder="Enter first number">

<input type="text" id="num2" placeholder="Enter second number">

<!-- Button to trigger the division operation -->

<button onclick="divideNumbers()">Divide</button>

<!-- Result and Error sections -->

<div id="result"></div>

<div id="error"></div>

</body>

</html>

1. A teacher has created a grade Labs function that verifies if student programming labs work. This function loops over an array of JavaScript objects that should contain a student property and run Lab property. The run Lab property is expected to be a function containing the student's code. The run Lab function is called and the result is compared to the expected result. If the result and expected result don't match, then the lab is considered a failure.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Grade Labs</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

button {

margin: 20px;

padding: 10px 20px;

font-size: 16px;

cursor: pointer;

}

#results {

margin-top: 30px;

font-size: 18px;

}

</style>

</head>

<body>

<h1>Student Lab Grading</h1>

<button onclick="gradeLabs()">Grade Labs</button>

<!-- Section to display the grading results -->

<div id="results"></div>

<script>

// Example array of student objects with runLab methods and expected results

const labs = [

{

student: 'Alice',

runLab: function () {

return 3 + 2; // Alice's solution

},

expectedResult: 5

},

{

student: 'Bob',

runLab: function () {

return 2 \* 2; // Bob's solution

},

expectedResult: 4

},

{

student: 'Charlie',

runLab: function () {

return 10 / 2; // Charlie's solution

},

expectedResult: 6 // Incorrect expected result for failure case

}

];

// Function to grade the labs and display the results

function gradeLabs() {

let resultDisplay = document.getElementById('results');

resultDisplay.innerHTML = ''; // Clear previous results

labs.forEach(lab => {

try {

const result = lab.runLab(); // Run the student's lab function

// Compare the result with the expected result

if (result === lab.expectedResult) {

resultDisplay.innerHTML += `<p style="color: green;">${lab.student}'s lab passed!</p>`;

} else {

throw new Error(`${lab.student}'s lab failed. Expected ${lab.expectedResult} but got ${result}.`);

}

} catch (error) {

resultDisplay.innerHTML += `<p style="color: red;">${error.message}</p>`;

}

});

}

</script>

</body>

</html>

1. There is an array of animals. The user is asked to enter the index for the animal they want to see.
   * If the user enters an index that does NOT contain an animal, the code will throw a Type Error when name is referenced on an undefined value.
   * Update the above program to print out the index the user entered. We want this message to be printed EVERY time the code runs

<html>

<head>

<title>Animal Array</title>

<script>

// Array of animals

const animals = ["Lion", "Tiger", "Elephant", "Giraffe", "Zebra"];

function showAnimal() {

// Get the value from the input field

const indexInput = document.getElementById("indexInput").value;

// Print the index entered by the user

document.getElementById("result").innerHTML = `You entered index: ${indexInput}`;

try {

// Convert the input to a number

const index = parseInt(indexInput, 10);

// Check if the input is a valid number

if (isNaN(index)) {

throw new Error("Invalid input. Please enter a number.");

}

// Check if the index is within the range of the array

if (index < 0 || index >= animals.length) {

throw new Error(`Index ${index} is out of range. Please enter a valid index between 0 and ${animals.length - 1}.`);

}

// If everything is valid, display the animal at the given index

document.getElementById("result").innerHTML += `<br>The animal at index ${index} is: ${animals[index]}`;

} catch (error) {

// Display the error message

document.getElementById("result").innerHTML += `<br>Error: ${error.message}`;

}

}

</script>

</head>

<body>

<h1>Animal Array</h1>

<p>Enter the index of the animal you want to see (0 to 4):</p>

<!-- Input field for the index -->

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

<!-- Button to submit the input -->

<button onclick="showAnimal()">Show Animal</button>

<!-- Div to display the result -->

<div id="result"></div>

</body>

</html>

1. Assume you have a function primitive Multiply that in 20 percent of cases multiplies two numbers and in the other 80 percent of cases raises an exception of type ***MultiplicatorUnitFailure***. Write a function that wraps this clunky function and just keeps trying until a call succeeds, after which it returns the result. Make sure you handle only the exceptions you are trying to handle

<html>

<head>

<title>Multiplication with Error Handling</title>

<script>

// Define the MultiplicatorUnitFailure exception class

class MultiplicatorUnitFailure extends Error {

constructor(message) {

super(message);

this.name = "MultiplicatorUnitFailure";

}

}

// Simulate the clunky primitiveMultiply function

function primitiveMultiply(a, b) {

if (Math.random() < 0.8) {

// In 80% of cases, throw an exception

throw new MultiplicatorUnitFailure("Multiplication failed, retrying...");

} else {

// In 20% of cases, return the product

return a \* b;

}

}

// A function that keeps trying until the multiplication succeeds

function reliableMultiply(a, b) {

while (true) {

try {

// Try multiplying using primitiveMultiply

return primitiveMultiply(a, b);

} catch (e) {

// Catch only MultiplicatorUnitFailure exceptions and retry

if (e instanceof MultiplicatorUnitFailure) {

console.log(e.message); // Log the error message to the console

} else {

// If it's another error, rethrow it

throw e;

}

}

}

}

// Function to test reliableMultiply when called by the user

function testMultiply() {

const num1 = parseInt(document.getElementById("num1").value, 10);

const num2 = parseInt(document.getElementById("num2").value, 10);

try {

const result = reliableMultiply(num1, num2);

document.getElementById("result").innerHTML = `Result: ${result}`;

} catch (error) {

document.getElementById("result").innerHTML = `Unexpected error: ${error.message}`;

}

}

</script>

</head>

<body>

<h1>Multiplication with Error Handling</h1>

<p>Enter two numbers to multiply:</p>

<!-- Input fields for two numbers -->

<input type="number" id="num1" placeholder="Number 1">

<input type="number" id="num2" placeholder="Number 2">

<!-- Button to trigger multiplication -->

<button onclick="testMultiply()">Multiply</button>

<!-- Div to display the result -->

<div id="result"></div>

</body>

</html>

**Exercises – 8 : JavaScript Classes**

1. Let us assume you need to organize a library of some electronic manuals and novels for a particular company. For each book, you need to store following information
   * The title
   * The author
   * The copyright date
   * The ISBN
   * The number of pages
   * The number of times the book has been checked out.
   * Whether the book has been discarded

Company wants to perform certain actions when the any book is outdated. The manual books must be exited after 5 years while a novel book should be exited if its checked out time crossed 100.

**Task 1.** Construct three classes that hold the information needed by company. One class should be a **Book** class and two child classes of the Book class called **Manual** and **Novel**. Each class will contain two methods. One will be a constructor. The other one will either be in charge of disposal of the book or updating the property related to the number of times a book has been checked out.

**Task 2:** Create an object of the Novel class and Manual class with some valid details.

**Task 3:** Write the method to count the duration of a manual book and checkout time for novel book so that they can be discarded based on the duration.

<html>

<head>

<title>Library Management</title>

<script>

// Parent Book class

class Book {

constructor(title, author, copyrightDate, ISBN, pages, timesCheckedOut, isDiscarded) {

this.title = title;

this.author = author;

this.copyrightDate = copyrightDate;

this.ISBN = ISBN;

this.pages = pages;

this.timesCheckedOut = timesCheckedOut;

this.isDiscarded = isDiscarded || false; // Default to false

}

dispose() {

console.log("Dispose method needs to be implemented in child class.");

}

}

// Manual class extends Book

class Manual extends Book {

constructor(title, author, copyrightDate, ISBN, pages, timesCheckedOut, isDiscarded) {

super(title, author, copyrightDate, ISBN, pages, timesCheckedOut, isDiscarded);

}

dispose() {

const currentYear = new Date().getFullYear();

const bookAge = currentYear - this.copyrightDate;

if (bookAge > 5) {

this.isDiscarded = true;

return `Manual "${this.title}" is discarded due to being more than 5 years old.`;

} else {

return `Manual "${this.title}" is still valid.`;

}

}

}

// Novel class extends Book

class Novel extends Book {

constructor(title, author, copyrightDate, ISBN, pages, timesCheckedOut, isDiscarded) {

super(title, author, copyrightDate, ISBN, pages, timesCheckedOut, isDiscarded);

}

dispose() {

if (this.timesCheckedOut > 100) {

this.isDiscarded = true;

return `Novel "${this.title}" is discarded due to being checked out more than 100 times.`;

} else {

return `Novel "${this.title}" is still valid.`;

}

}

checkOut() {

this.timesCheckedOut++;

}

}

// Function to handle manual book creation

function createManual() {

const title = document.getElementById("manualTitle").value;

const author = document.getElementById("manualAuthor").value;

const copyrightDate = parseInt(document.getElementById("manualCopyright").value, 10);

const ISBN = document.getElementById("manualISBN").value;

const pages = parseInt(document.getElementById("manualPages").value, 10);

const timesCheckedOut = parseInt(document.getElementById("manualCheckedOut").value, 10);

const manualBook = new Manual(title, author, copyrightDate, ISBN, pages, timesCheckedOut);

// Show whether the manual should be discarded or not

document.getElementById("manualResult").innerText = manualBook.dispose();

}

// Function to handle novel book creation

function createNovel() {

const title = document.getElementById("novelTitle").value;

const author = document.getElementById("novelAuthor").value;

const copyrightDate = parseInt(document.getElementById("novelCopyright").value, 10);

const ISBN = document.getElementById("novelISBN").value;

const pages = parseInt(document.getElementById("novelPages").value, 10);

const timesCheckedOut = parseInt(document.getElementById("novelCheckedOut").value, 10);

const novelBook = new Novel(title, author, copyrightDate, ISBN, pages, timesCheckedOut);

// Show whether the novel should be discarded or not

document.getElementById("novelResult").innerText = novelBook.dispose();

}

</script>

</head>

<body>

<h1>Library Management</h1>

<!-- Manual Book Section -->

<h2>Manual Book</h2>

<label>Title: <input type="text" id="manualTitle"></label><br>

<label>Author: <input type="text" id="manualAuthor"></label><br>

<label>Copyright Date: <input type="number" id="manualCopyright"></label><br>

<label>ISBN: <input type="text" id="manualISBN"></label><br>

<label>Pages: <input type="number" id="manualPages"></label><br>

<label>Times Checked Out: <input type="number" id="manualCheckedOut"></label><br>

<button onclick="createManual()">Check Manual Status</button>

<!-- Result for manual -->

<p id="manualResult"></p>

<hr>

<!-- Novel Book Section -->

<h2>Novel Book</h2>

<label>Title: <input type="text" id="novelTitle"></label><br>

<label>Author: <input type="text" id="novelAuthor"></label><br>

<label>Copyright Date: <input type="number" id="novelCopyright"></label><br>

<label>ISBN: <input type="text" id="novelISBN"></label><br>

<label>Pages: <input type="number" id="novelPages"></label><br>

<label>Times Checked Out: <input type="number" id="novelCheckedOut"></label><br>

<button onclick="createNovel()">Check Novel Status</button>

<!-- Result for novel -->

<p id="novelResult"></p>

</body>

</html>

1. Create an object ‘product’ with properties: ID, Name, Description and Price. Then create function for this object called displayProductDetails(). This function when invoked should display the name, description and discounted price of the product. For the calculation of discounted price, you need to create another function CalculateDisc(percentage). The discount percentage would be given by user.

<html>

<head>

<title>Product Details</title>

<script>

// Product class definition

class Product {

constructor(ID, name, description, price) {

this.ID = ID;

this.name = name;

this.description = description;

this.price = price;

}

// Method to calculate discounted price

calculateDisc(percentage) {

const discountAmount = (this.price \* percentage) / 100;

return this.price - discountAmount;

}

// Method to display product details

displayProductDetails(discountPercentage) {

const discountedPrice = this.calculateDisc(discountPercentage);

return `Product Name: ${this.name}<br>Description: ${this.description}<br>Discounted Price: $${discountedPrice.toFixed(2)}`;

}

}

// Function to create and display product details

function showProductDetails() {

// Create a Product object

const product = new Product(

101, // ID

"Smartphone", // Name

"Latest model with advanced features", // Description

699.99 // Price

);

// Get discount percentage from user input

const discountPercentage = parseFloat(document.getElementById("discount").value);

// Display product details

document.getElementById("productDetails").innerHTML = product.displayProductDetails(discountPercentage);

}

</script>

</head>

<body>

<h1>Product Details</h1>

<p>Enter Discount Percentage:</p>

<input type="number" id="discount" placeholder="Discount Percentage" min="0" max="100">

<button onclick="showProductDetails()">Show Product Details</button>

<!-- Div to display product details -->

<div id="productDetails"></div>

</body>

</html>

**Exercises 9: JavaScript Asynchronous programming**

1. Implement a clock in HTML which shows the live time in the format HH:MM:SS. Use the JavaScript setInterval() method and a callback function by displaying the system time every second.

<html>

<head>

<title>Live Clock</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 20%;

}

#clock {

font-size: 48px;

color: #333;

}

</style>

<script>

// Function to update the clock

function updateClock() {

const now = new Date(); // Get the current date and time

const hours = String(now.getHours()).padStart(2, '0'); // Get hours and format

const minutes = String(now.getMinutes()).padStart(2, '0'); // Get minutes and format

const seconds = String(now.getSeconds()).padStart(2, '0'); // Get seconds and format

// Display the time in HH:MM:SS format

document.getElementById("clock").innerText = `${hours}:${minutes}:${seconds}`;

}

// Start the clock when the page loads

window.onload = function() {

updateClock(); // Initial call to display time immediately

setInterval(updateClock, 1000); // Update time every second

};

</script>

</head>

<body>

<h1>Live Clock</h1>

<div id="clock"></div> <!-- Div to display the clock -->

</body>

</html>

1. Sometimes we use external resource files in our HTML page. The content of an external file cannot be used until loaded completely. This can be implemented with JavaScript Callback. Design a web page which loads an external HTML file on the 43 Web Design Lab Web Design Lab event: onLoad. Hint: use the inbuilt class XMLHttpRequest() to use its functions: open(), onLoad() etc. to load an external HTML file.

**external-content.html:**

<html>

<head>

<title>External Content</title>

</head>

<body>

<h2>External Content Loaded</h2>

<p>This content is loaded from an external HTML file.</p>

</body>

</html>

**Index.html**

<html>

<head>

<title>Load External HTML</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

#externalContent {

border: 1px solid #ccc;

padding: 20px;

margin-top: 20px;

display: inline-block;

}

</style>

<script>

// Function to load external HTML file

function loadExternalContent() {

const xhr = new XMLHttpRequest(); // Create a new XMLHttpRequest object

// Configure it: GET-request for the URL of the external file

xhr.open('GET', 'external-content.html', true);

// Set up the onload callback

xhr.onload = function() {

if (xhr.status === 200) { // Check if the request was successful

document.getElementById('externalContent').innerHTML = xhr.responseText; // Load the response into the div

} else {

document.getElementById('externalContent').innerHTML = 'Error loading content.';

}

};

// Send the request

xhr.send();

}

// Load external content on window load

window.onload = loadExternalContent;

</script>

</head>

<body>

<h1>Load External HTML File Example</h1>

<div id="externalContent">Loading external content...</div> <!-- Div to display the external content -->

</body>

</html>

1. The village crows own an old scalpel that they occasionally use on special missions²say, to cut through screen doors or packaging. To be able to quickly track it down, every time the scalpel is moved to another nest, an entry is added to the storage of both the nest that had it and the nest that took it, under the name "scalpel", with its new location as the value.This means that finding the scalpel is a matter of following the breadcrumb trail of storage entries, until you find a nest where that points at the nest itself.

Write an async function locateScalpel that does this, starting at the nest on which it runs. You can use the anyStorage function defined earlier to access storage in arbitrary nests. The scalpel has been going around long enough that you may assume that every nest has a "scalpel" entry in its data storage.

<html>

<head>

<title>Locate Scalpel</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

#result {

margin-top: 20px;

font-weight: bold;

}

</style>

<script>

// Simulated storage for nests

const nests = {

'Nest1': { scalpel: 'Nest2' },

'Nest2': { scalpel: 'Nest3' },

'Nest3': { scalpel: 'Nest1' },

'Nest4': { scalpel: 'Nest5' },

'Nest5': { scalpel: 'Nest4' }

};

// Simulate asynchronous storage access

function anyStorage(nestName) {

return new Promise((resolve) => {

setTimeout(() => {

resolve(nests[nestName]);

}, 500); // Simulate network delay

});

}

// Async function to locate the scalpel

async function locateScalpel(startNest) {

let currentNest = startNest;

const visitedNests = new Set();

while (true) {

// If we have already visited this nest, we've found the loop

if (visitedNests.has(currentNest)) {

return `Scalpel located at: ${currentNest} (loop detected)`;

}

// Mark the current nest as visited

visitedNests.add(currentNest);

// Access the storage of the current nest

const storage = await anyStorage(currentNest);

// Get the next nest from the storage

currentNest = storage.scalpel;

}

}

// Function to start the search from a specific nest

async function startSearch() {

const result = await locateScalpel('Nest1');

document.getElementById('result').innerText = result;

}

</script>

</head>

<body>

<h1>Locate the Scalpel</h1>

<button onclick="startSearch()">Locate Scalpel Starting from Nest1</button>

<div id="result">Click the button to start locating the scalpel.</div>

</body>

</html>