**II Year II Semester**

|  |  |  |  |
| --- | --- | --- | --- |
| **L** | **T** | **P** | **C** |
| **0** | **1** | **2** | **2** |

# FULL STACK DEVELOPMENT – 1 (SKILL ENHANCEMENT COURSE)

## Course Objectives:

The main objectives of the course are to

1. Make use of HTML elements and their attributes for designing static web pages
2. Build a web page by applying appropriate CSS styles to HTML elements
3. Experiment with JavaScript to develop dynamic web pages and validate forms

## Experiments covering the Topics:

* + Lists, Links and Images
  + HTML Tables, Forms and Frames
  + HTML 5 and Cascading Style Sheets, Types of CSS
  + Selector forms
  + CSS with Color, Background, Font, Text and CSS Box Model
  + Applying JavaScript - internal and external, I/O, Type Conversion
  + JavaScript Conditional Statements and Loops, Pre-defined and User-defined Objects
  + JavaScript Functions and Events
  + Node.js

## Sample Experiments:

1. **Lists, Links and Images**
   1. Write a HTML program, to explain the working of lists.

Note: It should have an ordered list, unordered list, nested lists and ordered list in an unordered list and definition lists.

* 1. \target Attributes.
  2. Create a HTML document that has your image and your friend’s image with a specific height and width. Also when clicked on the images it should navigate to their respective profiles.
  3. Write a HTML program, in such a way that, rather than placing large images on a page, the preferred technique is to use thumbnails by setting the height and width parameters to something like to 100\*100 pixels. Each thumbnail image is also a link to a full sized version of the image. Create an image gallery using this technique

## HTML Tables, Forms and Frames

* Write a HTML program, to explain the working of tables. (use tags: <table>, <tr>, <th>,

<td> and attributes: border, rowspan, colspan)

* Write a HTML program, to explain the working of tables by preparing a timetable. (Note: Use <caption> tag to set the caption to the table & also use cell spacing, cell padding, border, rowspan, colspan etc.).
* Write a HTML program, to explain the working of forms by designing Registration form. (Note: Include text field, password field, number field, date of birth field, checkboxes, radio buttons, list boxes using <select>&<option> tags, <text area> and two buttons ie: submit and reset. Use tables to provide a better view).
* Write a HTML program, to explain the working of frames, such that page is to be divided into 3 parts on either direction. (Note: first frame   image, second frame   paragraph, third frame   hyperlink. And also make sure of using “no frame” attribute such that frames to be fixed).

## HTML 5 and Cascading Style Sheets, Types of CSS

* 1. Write a HTML program, that makes use of <article>, <aside>, <figure>, <figcaption>,

<footer>, <header>, <main>, <nav>, <section>, <div>, <span> tags.

* 1. Write a HTML program, to embed audio and video into HTML web page.
  2. Write a program to apply different types (or levels of styles or style specification formats)

- inline, internal, external styles to HTML elements. (identify selector, property and value).

## Selector forms

* 1. Write a program to apply different types of selector forms
     + Simple selector (element, id, class, group, universal)
     + Combinator selector (descendant, child, adjacent sibling, general sibling)
     + Pseudo-class selector
     + Pseudo-element selector
     + Attribute selector

## CSS with Color, Background, Font, Text and CSS Box Model

* 1. Write a program to demonstrate the various ways you can reference a color in CSS.
  2. Write a CSS rule that places a background image halfway down the page, tilting it horizontally. The image should remain in place when the user scrolls up or down.
  3. Write a program using the following terms related to CSS font and text:

i. font-size ii. font-weight iii. font-style

iv. text-decoration v. text-transformation vi. text-alignment

* 1. Write a program, to explain the importance of CSS Box model using

i. Content ii. Border iii. Margin iv. padding

## Applying JavaScript - internal and external, I/O, Type Conversion

* 1. Write a program to embed internal and external JavaScript in a web page.
  2. Write a program to explain the different ways for displaying output.
  3. Write a program to explain the different ways for taking input.
  4. Create a webpage which uses prompt dialogue box to ask a voter for his name and age. Display the information in table format along with either the voter can vote or not

## JavaScript Pre-defined and User-defined Objects

* 1. Write a program using document object properties and methods.
  2. Write a program using window object properties and methods.
  3. Write a program using array object properties and methods.
  4. Write a program using math object properties and methods.
  5. Write a program using string object properties and methods.
  6. Write a program using regex object properties and methods.
  7. Write a program using date object properties and methods.
  8. Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.

## JavaScript Conditional Statements and Loops

* 1. Write a program which asks the user to enter three integers, obtains the numbers from the user and outputs HTML text that displays the larger number followed by the words “LARGER NUMBER” in an information message dialog. If the numbers are equal, output HTML text as “EQUAL NUMBERS”.
  2. Write a program to display week days using switch case.
  3. Write a program to print 1 to 10 numbers using for, while and do-while loops.
  4. Write aprogram to print data in object using for-in, for-each and for-of loops
  5. Develop a program to determine whether a given number is an ‘ARMSTRONG NUMBER’ or not. [Eg: 153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e.,13 + 53+ 33 = 153]
  6. Write a program to display the denomination of the amount deposited in the bank in terms of 100’s, 50’s, 20’s, 10’s, 5’s, 2’s & 1’s. (Eg: If deposited amount is Rs.163, the output should be 1-100’s, 1-50’s, 1- 10’s, 1-2’s & 1-1’s)

## Javascript Functions and Events

* 1. Design a appropriate function should be called to display
     + Factorial of that number
     + Fibonacci series up to that number
     + Prime numbers up to that number
     + Is it palindrome or not
  2. Design a HTML having a text box and four buttons named Factorial, Fibonacci, Prime, and Palindrome. When a button is pressed an appropriate function should be called to display

1. Factorial of that number
2. Fibonacci series up to that number
3. Prime numbers up to that number
4. Is it palindrome or not
   1. Write a program to validate the following fields in a registration page
5. Name (start with alphabet and followed by alphanumeric and the length should not be less than 6 characters)
6. Mobile (only numbers and length 10 digits)
7. E-mail (should contain format like xxxxxxx@xxxxxx.xxx)

## Text Books:

1. Programming the World Wide Web, 7th Edition, Robet W Sebesta, Pearson, 2013.
2. Web Programming with HTML5, CSS and JavaScript, John Dean, Jones & Bartlett Learning, 2019 (Chapters 1-11).
3. Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node, Vasan Subramanian, 2nd edition, APress, O’Reilly.

## Web Links:

* 1. https://[www.w3schools.com/html](http://www.w3schools.com/html)
  2. https://[www.w3schools.com/css](http://www.w3schools.com/css)
  3. https://[www.w3schools.com/js/](http://www.w3schools.com/js/)
  4. https://[www.w3schools.com/nodejs](http://www.w3schools.com/nodejs)

https://[www.w3schools.com/typescript](http://www.w3schools.com/typescript)

1. **Lists, Links and Images**
   1. Write a HTML program, to explain the working of lists.

Note: It should have an ordered list, unordered list, nested lists and ordered list in an unordered list and definition lists.

This program demonstrates various types of lists in HTML:

* **Ordered List (<ol>)**: Numbered list items.
* **Unordered List (<ul>)**: Bullet points.
* **Nested Lists**: A list inside another list.
* **Ordered List inside an Unordered List**
* **Definition List (<dl>):** Term and description format.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Lists in HTML</title>

</head>

<body>

<h2>Ordered List Example</h2>

<ol>

<li>HTML</li>

<li>CSS</li>

<li>JavaScript</li>

</ol>

<h2>Unordered List Example</h2>

<ul>

<li>Apple</li>

<li>Banana</li>

<li>Grapes</li>

</ul>

<h2>Nested List Example</h2>

<ul>

<li>Fruits

<ul>

<li>Apple</li>

<li>Mango</li>

</ul>

</li>

<li>Vegetables

<ul>

<li>Carrot</li>

<li>Potato</li>

</ul>

</li>

</ul>

<h2>Ordered List inside an Unordered List</h2>

<ul>

<li>Frontend Technologies:

<ol>

<li>HTML</li>

<li>CSS</li>

<li>JavaScript</li>

</ol>

</li>

<li>Backend Technologies:

<ol>

<li>Node.js</li>

<li>Express.js</li>

</ol>

</li>

</ul>

<h2>Definition List Example</h2>

<dl>

<dt>HTML</dt>

<dd>HyperText Markup Language</dd>

<dt>CSS</dt>

<dd>Cascading Style Sheets</dd>

</dl>

</body>

</html>

* 1. \target Attributes.

The target attribute specifies how a linked document should be opened.

| **Attribute** | **Description** |
| --- | --- |
| \_self | Default. Opens in the same tab |
| \_blank | Opens in a new tab |
| \_parent | Opens in the parent frame |
| \_top | Opens in the full body of the window |

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Target Attribute Example</title>

</head>

<body>

<h2>Target Attribute in Links</h2>

<ul>

<li><a href="https://www.google.com" target="\_self">Google (Same Tab)</a></li>

<li><a href="https://www.google.com" target="\_blank">Google (New Tab)</a></li>

</ul>

</body>

</html>

* 1. Create a HTML document that has your image and your friend’s image with a specific height and width. Also when clicked on the images it should navigate to their respective profiles.

This HTML program displays two images (your and your friend's), which navigate to respective profiles when clicked.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Image Links</title>

</head>

<body>

<h2>My Profile</h2>

<a href="https://www.linkedin.com/in/yourprofile" target="\_blank">

<img src="your-image.jpg" alt="Your Image" width="150" height="150">

</a>

<h2>Friend's Profile</h2>

<a href="https://www.linkedin.com/in/friendprofile" target="\_blank">

<img src="friend-image.jpg" alt="Friend's Image" width="150" height="150">

</a>

</body>

</html>

* 1. Write a HTML program, in such a way that, rather than placing large images on a page, the preferred technique is to use thumbnails by setting the height and width parameters to something like to 100\*100 pixels. Each thumbnail image is also a link to a full sized version of the image. Create an image gallery using this technique

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Thumbnail Image Gallery</title>

</head>

<body>

<h2>Image Gallery</h2>

<a href="image1\_large.jpg" target="\_blank">

<img src="image1\_thumbnail.jpg" alt="Image 1" width="100" height="100">

</a>

<a href="image2\_large.jpg" target="\_blank">

<img src="image2\_thumbnail.jpg" alt="Image 2" width="100" height="100">

</a>

<a href="image3\_large.jpg" target="\_blank">

<img src="image3\_thumbnail.jpg" alt="Image 3" width="100" height="100">

</a>

</body>

</html>

## HTML Tables, Forms and Frames

* Write a HTML program, to explain the working of tables. (use tags: <table>, <tr>, <th>,

<td> and attributes: border, rowspan, colspan)

<!DOCTYPE html>

<html>

<head>

<title>HTML Tables</title>

</head>

<body>

<h2>Simple Table</h2>

<table border="1">

<tr>

<th>Header 1</th>

<th>Header 2</th>

</tr>

<tr>

<td>Row 1, Cell 1</td>

<td>Row 1, Cell 2</td>

</tr>

<tr>

<td colspan="2">Row 2, Merged Cell</td>

</tr>

</table>

</body>

</html>

* Write a HTML program, to explain the working of tables by preparing a timetable. (Note: Use <caption> tag to set the caption to the table & also use cell spacing, cell padding, border, rowspan, colspan etc.).

<!DOCTYPE html>

<html>

<head>

<title>Timetable</title>

</head>

<body>

<h2>Class Timetable</h2>

<table border="1" cellspacing="5" cellpadding="10">

<caption>Weekly Class Schedule</caption>

<tr>

<th>Day</th>

<th>9 AM - 10 AM</th>

<th>10 AM - 11 AM</th>

</tr>

<tr>

<td>Monday</td>

<td rowspan="2">Math</td>

<td>English</td>

</tr>

<tr>

<td>Tuesday</td>

<td>Science</td>

</tr>

</table>

</body>

</html>

* Write a HTML program, to explain the working of forms by designing Registration form. (Note: Include text field, password field, number field, date of birth field, checkboxes, radio buttons, list boxes using <select>&<option> tags, <text area> and two buttons ie: submit and reset. Use tables to provide a better view).

<!DOCTYPE html>

<html>

<head>

<title>Registration Form</title>

</head>

<body>

<h2>Registration Form</h2>

<form>

<table>

<tr>

<td>Name:</td>

<td><input type="text" name="name"></td>

</tr>

<tr>

<td>Password:</td>

<td><input type="password" name="password"></td>

</tr>

<tr>

<td>Gender:</td>

<td>

<input type="radio" name="gender" value="Male"> Male

<input type="radio" name="gender" value="Female"> Female

</td>

</tr>

<tr>

<td>Country:</td>

<td>

<select>

<option>India</option>

<option>USA</option>

</select>

</td>

</tr>

<tr>

<td>Comments:</td>

<td><textarea></textarea></td>

</tr>

<tr>

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

<td><input type="reset" value="Reset"></td>

</tr>

</table>

</form>

</body>

</html>

* Write a HTML program, to explain the working of frames, such that page is to be divided into 3 parts on either direction. (Note: first frame   image, second frame   paragraph, third frame   hyperlink. And also make sure of using “no frame” attribute such that frames to be fixed).

<!DOCTYPE html>

<html>

<head>

<title>Frames Example</title>

</head>

<frameset cols="33%, 33%, 34%">

<frame src="image.html">

<frame src="paragraph.html">

<frame src="link.html">

<noframes>

<body>Your browser does not support frames.</body>

</noframes>

</frameset>

</html>

<!-- image.html -->

<!DOCTYPE html>

<html>

<body>

<img src="example.jpg" alt="Example Image" width="100%" height="100%">

</body>

</html>

<!-- paragraph.html -->

<!DOCTYPE html>

<html>

<body>

<p>This is a sample paragraph displayed in the second frame.</p>

</body>

</html>

<!-- link.html -->

<!DOCTYPE html>

<html>

<body>

<a href="https://example.com" target="\_blank">Visit Example Website</a>

</body>

</html>

## HTML 5 and Cascading Style Sheets, Types of CSS

* 1. Write a HTML program, that makes use of <article>, <aside>, <figure>, <figcaption>,

<footer>, <header>, <main>, <nav>, <section>, <div>, <span> tags.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Semantic Tags Example</title>

<style>

body { font-family: Arial, sans-serif; margin: 0; padding: 0; }

header, nav, footer { background: #333; color: white; text-align: center; padding: 10px; }

main { padding: 20px; }

section { margin-bottom: 20px; }

article { border: 1px solid #ccc; padding: 10px; }

aside { background: #f4f4f4; padding: 10px; margin-top: 10px; }

figure { text-align: center; }

figure img { width: 200px; }

</style>

</head>

<body>

<header>

<h1>Welcome to My Website</h1>

</header>

<nav>

<a href="#">Home</a> | <a href="#">About</a> | <a href="#">Contact</a>

</nav>

<main>

<section>

<h2>Introduction</h2>

<p>This is an example of using semantic HTML elements.</p>

</section>

<section>

<h2>Articles and Figures</h2>

<article>

<h3>Article Title</h3>

<p>This is a sample article with an image.</p>

<figure>

<img src="https://via.placeholder.com/200" alt="Sample Image">

<figcaption>Sample Image with Caption</figcaption>

</figure>

</article>

</section>

<aside>

<h3>Sidebar Content</h3>

<p>This is some additional content in an aside section.</p>

</aside>

</main>

<footer>

<p>&copy; 2025 My Website</p>

</footer>

</body>

</html>

* 1. Write a HTML program, to embed audio and video into HTML web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Embedding Audio and Video</title>

</head>

<body>

<h2>Audio Embedding</h2>

<audio controls>

<source src="assets/sample-audio.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

<h2>Video Embedding</h2>

<video width="400" controls>

<source src="assets/sample-video.mp4" type="video/mp4">

Your browser does not support the video tag.

</video>

</body>

</html>

* 1. Write a program to apply different types (or levels of styles or style specification formats)

- inline, internal, external styles to HTML elements. (identify selector, property and value).

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Styles Example</title>

<link rel="stylesheet" href="styles/external.css"> <!-- External CSS -->

<style>

/\* Internal CSS \*/

.internal-style {

color: blue;

font-size: 20px;

}

</style>

</head>

<body>

<h1 style="color: red;">Inline CSS Example</h1> <!-- Inline CSS -->

<p class="internal-style">This paragraph uses internal CSS.</p>

<p class="external-style">This paragraph uses external CSS.</p>

</body>

</html>

.external-style {

color: green;

font-weight: bold;

}

## Selector forms

* 1. Write a program to apply different types of selector forms
     + Simple selector (element, id, class, group, universal)
     + Combinator selector (descendant, child, adjacent sibling, general sibling)
     + Pseudo-class selector
     + Pseudo-element selector
     + Attribute selector

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Selectors</title>

<style>

/\* Simple Selectors \*/

h1 { color: blue; }

#unique { color: green; }

.group { font-style: italic; }

/\* Combinator Selectors \*/

div > p { color: red; } /\* Child Selector \*/

div p { color: orange; } /\* Descendant Selector \*/

/\* Pseudo-class Selector \*/

a:hover { color: purple; }

/\* Pseudo-element Selector \*/

p::first-letter { font-size: 2em; color: brown; }

/\* Attribute Selector \*/

input[type="text"] { border: 2px solid blue; }

</style>

</head>

<body>

<h1>Simple Selectors</h1>

<p id="unique">This is an ID selector.</p>

<p class="group">This is a class selector.</p>

<div>

<p>This is inside a div.</p>

</div>

<a href="#">Hover over me</a>

<p>This paragraph demonstrates a pseudo-element.</p>

<input type="text" placeholder="Attribute Selector Example">

</body>

</html>

## CSS with Color, Background, Font, Text and CSS Box Model

* 1. Write a program to demonstrate the various ways you can reference a color in CSS.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Box Model</title>

<style>

.box {

width: 200px;

padding: 20px;

border: 5px solid blue;

margin: 10px;

background-color: lightgray;

}

</style>

</head>

<body>

<div class="box">This is a box model example.</div>

</body>

</html>

* 1. Write a CSS rule that places a background image halfway down the page, tilting it horizontally. The image should remain in place when the user scrolls up or down.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background Image Position</title>

<style>

body {

height: 200vh; /\* Makes the page scrollable \*/

background-image: url('assets/background.jpg');

background-repeat: no-repeat;

background-size: cover;

background-position: center 50%;

background-attachment: fixed; /\* Ensures the background stays in place \*/

transform: scaleX(-1); /\* Flips the image horizontally \*/

}

.content {

text-align: center;

font-size: 24px;

margin-top: 50vh; /\* Places content below the image \*/

color: white;

font-weight: bold;

}

</style>

</head>

<body>

<div class="content">This text appears below the background image.</div>

</body>

</html>

* 1. Write a program using the following terms related to CSS font and text:

i. font-size ii. font-weight iii. font-style

iv. text-decoration v. text-transformation vi. text-alignment

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background Image Position</title>

<style>

body {

height: 200vh; /\* Makes the page scrollable \*/

background-image: url('assets/background.jpg');

background-repeat: no-repeat;

background-size: cover;

background-position: center 50%;

background-attachment: fixed; /\* Ensures the background stays in place \*/

transform: scaleX(-1); /\* Flips the image horizontally \*/

}

.content {

text-align: center;

font-size: 24px;

margin-top: 50vh; /\* Places content below the image \*/

color: white;

font-weight: bold;

}

</style>

</head>

<body>

<div class="content">This text appears below the background image.</div>

</body>

</html>

* 1. Write a program, to explain the importance of CSS Box model using

i. Content ii. Border iii. Margin iv. Padding

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Box Model</title>

<style>

.box {

width: 300px;

padding: 20px; /\* Space inside the box \*/

border: 5px solid blue; /\* Border surrounding the content \*/

margin: 30px auto; /\* Space outside the box \*/

background-color: lightgray;

text-align: center;

}

</style>

</head>

<body>

<div class="box">

<p>This box demonstrates the CSS Box Model.</p>

</div>

</body>

</html>

## Applying JavaScript - internal and external, I/O, Type Conversion

* 1. Write a program to embed internal and external JavaScript in a web page.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>JavaScript Example</title>

<script src="scripts/external.js"></script>

<script>

function internalJS() {

alert("This is internal JavaScript!");

}

</script>

</head>

<body>

<button onclick="internalJS()">Internal JS</button>

<button onclick="externalJS()">External JS</button>

</body>

</html>

function externalJS() {

alert("This is external JavaScript!");

}

* 1. Write a program to explain the different ways for displaying output.
  2. Write a program to explain the different ways for taking input.
  3. Create a webpage which uses prompt dialogue box to ask a voter for his name and age. Display the information in table format along with either the voter can vote or not

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Voter Verification</title>

</head>

<body>

<script>

let name = prompt("Enter your name:");

let age = prompt("Enter your age:");

let canVote = age >= 18 ? "Yes" : "No";

document.write(`

<h2>Voter Details</h2>

<table border="1">

<tr><th>Name</th><td>${name}</td></tr>

<tr><th>Age</th><td>${age}</td></tr>

<tr><th>Eligible to Vote</th><td>${canVote}</td></tr>

</table>

`);

</script>

</body>

</html>

## JavaScript Pre-defined and User-defined Objects

* 1. Write a program using document object properties and methods.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Document Object</title>

</head>

<body>

<h1 id="heading">Hello World</h1>

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

<script>

function changeText() {

document.getElementById("heading").innerText = "Text Changed!";

alert("Title: " + document.title + "\nURL: " + document.URL);

}

</script>

</body>

</html>

* 1. Write a program using window object properties and methods.

<!DOCTYPE html>

<html lang="en">

<head>

<title>Window Object</title>

</head>

<body>

<button onclick="openWindow()">Open New Window</button>

<button onclick="closeWindow()">Close Window</button>

<script>

let newWin;

function openWindow() {

newWin = window.open("", "", "width=300,height=200");

newWin.document.write("<p>New Window Opened!</p>");

}

function closeWindow() {

if (newWin) newWin.close();

}

</script>

</body>

</html>

* 1. Write a program using array object properties and methods.

<script>

let numbers = [10, 20, 30, 40, 50];

console.log("Original Array:", numbers);

numbers.push(60);

console.log("After Push:", numbers);

numbers.pop();

console.log("After Pop:", numbers);

console.log("Array Length:", numbers.length);

</script>

* 1. Write a program using math object properties and methods.

<script>

console.log("Math.PI:", Math.PI);

console.log("Random Number:", Math.random());

console.log("Square Root of 16:", Math.sqrt(16));

console.log("Power (2^5):", Math.pow(2, 5));

</script>

* 1. Write a program using string object properties and methods.

<script>

let str = "JavaScript is Awesome!";

console.log("Original String:", str);

console.log("Length:", str.length);

console.log("Uppercase:", str.toUpperCase());

console.log("Substring(0, 10):", str.substring(0, 10));

</script>

* 1. Write a program using regex object properties and methods.

<script>

let str = "Hello 123 JavaScript!";

let regex = /\d+/g; // Find digits

console.log("Digits in String:", str.match(regex));

</script>

* 1. Write a program using date object properties and methods.

<script>

let today = new Date();

console.log("Current Date:", today);

console.log("Year:", today.getFullYear());

console.log("Month:", today.getMonth() + 1);

console.log("Day:", today.getDate());

</script>

* 1. Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.

<script>

function Car(model, year) {

this.model = model;

this.year = year;

this.getDetails = function () {

return `Car Model: ${this.model}, Year: ${this.year}`;

};

}

let myCar = new Car("Tesla", 2022);

console.log(myCar.getDetails());

</script>

## JavaScript Conditional Statements and Loops

* 1. Write a program which asks the user to enter three integers, obtains the numbers from the user and outputs HTML text that displays the larger number followed by the words “LARGER NUMBER” in an information message dialog. If the numbers are equal, output HTML text as “EQUAL NUMBERS”.

<script>

let a = parseInt(prompt("Enter first number:"));

let b = parseInt(prompt("Enter second number:"));

let c = parseInt(prompt("Enter third number:"));

let largest = Math.max(a, b, c);

if (a === b && b === c) {

document.write("EQUAL NUMBERS");

} else {

document.write(largest + " LARGER NUMBER");

}

</script>

* 1. Write a program to display week days using switch case.

<script>

let day = parseInt(prompt("Enter a number (1-7):"));

switch (day) {

case 1: alert("Sunday"); break;

case 2: alert("Monday"); break;

case 3: alert("Tuesday"); break;

case 4: alert("Wednesday"); break;

case 5: alert("Thursday"); break;

case 6: alert("Friday"); break;

case 7: alert("Saturday"); break;

default: alert("Invalid Input");

}

</script>

* 1. Write a program to print 1 to 10 numbers using for, while and do-while loops.

<script>

console.log("Using for loop:");

for (let i = 1; i <= 10; i++) console.log(i);

console.log("Using while loop:");

let j = 1;

while (j <= 10) console.log(j++);

console.log("Using do-while loop:");

let k = 1;

do {

console.log(k++);

} while (k <= 10);

</script>

* 1. Write aprogram to print data in object using for-in, for-each and for-of loops

<script>

let person = { name: "John", age: 25, city: "New York" };

console.log("Using for-in:");

for (let key in person) console.log(key + ":", person[key]);

console.log("Using for-of:");

let arr = ["Apple", "Banana", "Cherry"];

for (let item of arr) console.log(item);

console.log("Using forEach:");

arr.forEach(item => console.log(item));

</script>

* 1. Develop a program to determine whether a given number is an ‘ARMSTRONG NUMBER’ or not. [Eg: 153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e.,13 + 53+ 33 = 153]

<script>

let num = parseInt(prompt("Enter a number:"));

let sum = 0, temp = num;

while (temp > 0) {

let digit = temp % 10;

sum += digit \*\* 3;

temp = Math.floor(temp / 10);

}

alert(num === sum ? "Armstrong Number" : "Not an Armstrong Number");

</script>

* 1. Write a program to display the denomination of the amount deposited in the bank in terms of 100’s, 50’s, 20’s, 10’s, 5’s, 2’s & 1’s. (Eg: If deposited amount is Rs.163, the output should be 1-100’s, 1-50’s, 1- 10’s, 1-2’s & 1-1’s)

<script>

let amount = parseInt(prompt("Enter amount:"));

let denominations = [100, 50, 20, 10, 5, 2, 1];

let result = "";

for (let note of denominations) {

if (amount >= note) {

let count = Math.floor(amount / note);

amount %= note;

result += `${count} x Rs.${note}<br>`;

}

}

document.write(result);

</script>

## Javascript Functions and Events

* 1. Design a appropriate function should be called to display
     + Factorial of that number
     + Fibonacci series up to that number
     + Prime numbers up to that number
     + Is it palindrome or not

<script>

// Function to calculate factorial

function factorial(n) {

if (n === 0 || n === 1) return 1;

return n \* factorial(n - 1);

}

// Function to generate Fibonacci series

function fibonacci(n) {

let fib = [0, 1];

for (let i = 2; i < n; i++) {

fib[i] = fib[i - 1] + fib[i - 2];

}

return fib.slice(0, n);

}

// Function to check for prime numbers

function primeNumbers(n) {

let primes = [];

for (let i = 2; i <= n; i++) {

let isPrime = true;

for (let j = 2; j <= Math.sqrt(i); j++) {

if (i % j === 0) {

isPrime = false;

break;

}

}

if (isPrime) primes.push(i);

}

return primes;

}

// Function to check palindrome

function isPalindrome(n) {

let str = n.toString();

return str === str.split("").reverse().join("");

}

// Testing

let num = parseInt(prompt("Enter a number:"));

console.log("Factorial:", factorial(num));

console.log("Fibonacci:", fibonacci(num));

console.log("Prime numbers:", primeNumbers(num));

console.log("Palindrome:", isPalindrome(num));

</script>

* 1. Design a HTML having a text box and four buttons named Factorial, Fibonacci, Prime, and Palindrome. When a button is pressed an appropriate function should be called to display

1. Factorial of that number
2. Fibonacci series up to that number
3. Prime numbers up to that number
4. Is it palindrome or not

<!DOCTYPE html>

<html lang="en">

<head>

<title>JS Functions</title>

</head>

<body>

<h2>Enter a number:</h2>

<input type="number" id="num">

<button onclick="showFactorial()">Factorial</button>

<button onclick="showFibonacci()">Fibonacci</button>

<button onclick="showPrime()">Prime</button>

<button onclick="showPalindrome()">Palindrome</button>

<p id="result"></p>

<script>

function getValue() {

return parseInt(document.getElementById("num").value);

}

function showFactorial() {

let num = getValue();

document.getElementById("result").innerText = "Factorial: " + factorial(num);

}

function showFibonacci() {

let num = getValue();

document.getElementById("result").innerText = "Fibonacci: " + fibonacci(num).join(", ");

}

function showPrime() {

let num = getValue();

document.getElementById("result").innerText = "Prime Numbers: " + primeNumbers(num).join(", ");

}

function showPalindrome() {

let num = getValue();

document.getElementById("result").innerText = isPalindrome(num) ? "Palindrome" : "Not a Palindrome";

}

function factorial(n) {

if (n === 0 || n === 1) return 1;

return n \* factorial(n - 1);

}

function fibonacci(n) {

let fib = [0, 1];

for (let i = 2; i < n; i++) {

fib[i] = fib[i - 1] + fib[i - 2];

}

return fib.slice(0, n);

}

function primeNumbers(n) {

let primes = [];

for (let i = 2; i <= n; i++) {

let isPrime = true;

for (let j = 2; j <= Math.sqrt(i); j++) {

if (i % j === 0) {

isPrime = false;

break;

}

}

if (isPrime) primes.push(i);

}

return primes;

}

function isPalindrome(n) {

let str = n.toString();

return str === str.split("").reverse().join("");

}

</script>

</body>

</html>

* 1. Write a program to validate the following fields in a registration page

1. Name (start with alphabet and followed by alphanumeric and the length should not be less than 6 characters)
2. Mobile (only numbers and length 10 digits)
3. E-mail (should contain format like xxxxxxx@xxxxxx.xxx)

<!DOCTYPE html>

<html lang="en">

<head>

<title>Registration Form</title>

<script>

function validateForm() {

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

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

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

let namePattern = /^[A-Za-z][A-Za-z0-9]{5,}$/;

let mobilePattern = /^[0-9]{10}$/;

let emailPattern = /^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;

if (!namePattern.test(name)) {

alert("Invalid Name! It should start with an alphabet and be at least 6 characters long.");

return false;

}

if (!mobilePattern.test(mobile)) {

alert("Invalid Mobile Number! It should contain exactly 10 digits.");

return false;

}

if (!emailPattern.test(email)) {

alert("Invalid Email! It should follow the format xxxxxxx@xxxxxx.xxx");

return false;

}

alert("Form Submitted Successfully!");

return true;

}

</script>

</head>

<body>

<h2>Registration Form</h2>

<form onsubmit="return validateForm()">

<label>Name:</label>

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

<label>Mobile:</label>

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

<label>Email:</label>

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

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

</form>

</body>

</html>