

Full Stack Development-1

1. Lists, Links and Images

A. 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.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>HTML Lists Example</title>
</head>
<body>
<h1>Working of Lists in HTML</h1>

<h2>1. Ordered List</h2>
<ol>
<li>First item</li>
<li>Second item</li>
<li>Third item</li>
</ol>

<h2>2. Unordered List</h2>
<ul>
<li>Apple</li>
<li>Banana</li>
<li>Cherry</li>
</ul>

<h2>3. Nested Lists</h2>
<ul>
<li>Fruits
<ul>
<li>Apple</li>
<li>Banana</li>
</ul>
</li>
<li>Vegetables
<ul>
<li>Carrot</li>
<li>Broccoli</li>
</ul>
</li>
</ul>
```

```
</li>
</ul>
```

<h2>4. Ordered List inside an Unordered List</h2>

```
<ul>
<li>Steps to make Tea:
<ol>
<li>Boil water</li>
<li>Add tea leaves</li>
<li>Simmer</li>
<li>Strain and serve</li>
</ol>
</li>
</ul>
```

<h2>5. Definition List</h2>

```
<dl>
<dt>HTML</dt>
<dd>HyperText Markup Language - standard markup language for creating web
pages.</dd>
<dt>CSS</dt>
<dd>Cascading Style Sheets - used for styling HTML content.</dd>
<dt>JavaScript</dt>
<dd>Programming language that adds interactivity to web pages.</dd>
</dl>
</body>
</html>
```

Output:



1. Ordered List

1. First item
2. Second item
3. Third item

2. Unordered List

- Apple
- Banana
- Cherry

3. Nested Lists

- Fruits
 - Apple
 - Banana
- Vegetables
 - Carrot
 - Broccoli

4. Ordered List inside an Unordered List

- Steps to make Tea:
 1. Boil water
 2. Add tea leaves
 3. Simmer
 4. Strain and serve

5. Definition List

HTML
HyperText Markup Language - standard markup language for creating web pages.

CSS
Cascading Style Sheets - used for styling HTML content.

JavaScript
Programming language that adds interactivity to web pages.

1B. . Write a HTML program, to explain the working of hyperlinks using tag and href, target Attributes.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Hyperlink Example</title>
</head>
<body>

<h1>HTML Hyperlink Demonstration</h1>

<p>This is an example of using the <code>&lt;a&gt;</code> tag with
<code>href</code> and <code>target</code> attributes.</p>

<!-- Hyperlink that opens in the same tab -->
<p>
<a href="https://www.example.com" target="_self">Visit Example.com (same
tab)</a>
</p>

<!-- Hyperlink that opens in a new tab -->
<p>
<a href="https://www.wikipedia.org" target="_blank">Visit Wikipedia (new tab)</a>
</p>

<!-- Hyperlink that opens in the parent frame (used with iframes) -->
<p>
<a href="https://www.openai.com" target="_parent">Visit OpenAI (parent
frame)</a>
</p>

<!-- Hyperlink that opens in the full body of the window (used with framesets) -->
<p>
<a href="https://www.google.com" target="_top">Visit Google (full window)</a>
</p>

</body>
</html>
```

OutPut:

HTML Hyperlink Demonstration

This is an example of using the <a> tag with href and target attributes.

[Visit Example.com \(same tab\)](#)

[Visit Wikipedia \(new tab\)](#)

[Visit OpenAI \(parent frame\)](#)

[Visit Google \(full window\)](#)

1C. 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.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Profile Links</title>
</head>
<body>

<h1>Profiles</h1>

<table border="1" cellpadding="10">
<tr>
<!-- Your Profile -->
<td align="center">
<a href="https://www.yourprofile.com" target="_blank">
<br>
<strong>My Profile</strong>
</a>
</td>

<!-- Friend's Profile -->
<td align="center">
```

```

<a
href="https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.pexels.com%2
Fsearch%2Fprofile%2520picture%2F&psig=AOvVaw3Q_LyvF_3BwdaGb8WKhQ0k&us
t=1747108060459000&source=images&cd=vfe&opi=89978449&ved=0CBUQjhxqFwo
TCKDumeyCnY0DFQAAAAAdAAAAABAE" target="_blank">
<br>
<strong>Friend's Profile</strong>
</a>
</td>
</tr>
</table>
</body>
</html>

```

Output:

Profiles



1D. 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

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Thumbnail Image Gallery</title>
</head>
<body>
<h1>Image Gallery</h1>
<p>Click on a thumbnail to view the full-sized image.</p>
<div style="display: flex; gap: 15px; flex-wrap: wrap;">
<!-- Image 1 -->
<a href="https://www.pexels.com/photo/brown-giraffe-walking-on-brown-grass-
67552/" target="_blank">


```

```
</a>
```

```
<!-- Image 2 -->
```

```
<a href="https://www.pexels.com/photo/close-up-of-a-siamese-fighting-fish-325045/" target="_blank">
```

```

```

```
</a>
```

```
<!-- Image 3 -->
```

```
<a href="https://www.pexels.com/photo/black-and-green-toucan-on-tree-branch-17811/" target="_blank">
```

```

```

```
</a>
```

```
<!-- Add more images as needed -->
```

```
</div>
```

```
</body>
```

```
</html>
```

Output:

Image Gallery

Click on a thumbnail to view the full-sized image.

[Image 1](#) [Image 2](#) [Image 3](#)

2.A.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)

Program:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>HTML Table Example</title>
```

```
</head>
```

```
<body>
```

```
<h1>HTML Table Demonstration</h1>
```

```
<table border="1">
```

```
<tr>
```

```

<th rowspan="2">Name</th>
<th colspan="2">Marks</th>
<th rowspan="2">Total</th>
</tr>
<tr>
<th>Math</th>
<th>Science</th>
</tr>
<tr>
<td>Alice</td>
<td>85</td>
<td>90</td>
<td>175</td>
</tr>
<tr>
<td>Bob</td>
<td>78</td>
<td>88</td>
<td>166</td>
</tr>
<tr>
<td>Charlie</td>
<td>92</td>
<td>95</td>
<td>187</td>
</tr>
</table>

```

```

</body>
</html>

```

Output:

HTML Table Demonstration

Name	Marks		Total
	Math	Science	
Alice	85	90	175
Bob	78	88	166
Charlie	92	95	187

2b. Write a HTML program, to explain the working of tables by preparing a timetable. (Note: Use tag to set the caption to the table & also use cell spacing, cell padding, border, rowspan, colspan etc.).

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Class Timetable</title>
</head>
<body>

<h1>School Class Timetable</h1>

<table border="1" cellspacing="5" cellpadding="10">
<caption><strong>Weekly Class Timetable</strong></caption>
<tr>
<th>Day</th>
<th>Period 1<br>9:00 - 10:00</th>
<th>Period 2<br>10:00 - 11:00</th>
<th>Break<br>11:00 - 11:30</th>
<th>Period 3<br>11:30 - 12:30</th>
<th>Period 4<br>12:30 - 1:30</th>
</tr>

<tr>
<td>Monday</td>
<td>Math</td>
<td>Science</td>
<td rowspan="5" align="center">Break</td>
<td>English</td>
<td>History</td>
</tr>

<tr>
<td>Tuesday</td>
<td>English</td>
<td>Math</td>
<td>Geography</td>
<td>Science</td>
</tr>

<tr>
<td>Wednesday</td>
<td colspan="2" align="center">Project Work</td>
```



```

<td>Math</td>
<td>Art</td>
</tr>

<tr>
<td>Thursday</td>
<td>Science</td>
<td>English</td>
<td>History</td>
<td>Sports</td>
</tr>

<tr>
<td>Friday</td>
<td>Geography</td>
<td>Computer</td>
<td>Math</td>
<td>Music</td>
</tr>
</table>

</body>
</html>

```

Output:

School Class Timetable

Weekly Class Timetable					
Day	Period 1 9:00 - 10:00	Period 2 10:00 - 11:00	Break 11:00 - 11:30	Period 3 11:30 - 12:30	Period 4 12:30 - 1:30
Monday	Math	Science	Break	English	History
Tuesday	English	Math		Geography	Science
Wednesday	Project Work			Math	Art
Thursday	Science	English		History	Sports
Friday	Geography	Computer		Math	Music

2C. 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 and two buttons ie: submit and reset. Use tables to provide a better view).

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Registration Form</title>
</head>
<body>

<h1>User Registration Form</h1>

<form action="#" method="post">
<table border="1" cellpadding="10" cellspacing="0">
<caption><strong>Registration Details</strong></caption>

<!-- Name -->
<tr>
<td><label for="fullname">Full Name:</label></td>
<td><input type="text" id="fullname" name="fullname" required></td>
</tr>

<!-- Password -->
<tr>
<td><label for="password">Password:</label></td>
<td><input type="password" id="password" name="password" required></td>
</tr>

<!-- Age -->
<tr>
<td><label for="age">Age:</label></td>
<td><input type="number" id="age" name="age" min="1" max="100"></td>
</tr>

<!-- Date of Birth -->
<tr>
<td><label for="dob">Date of Birth:</label></td>
<td><input type="date" id="dob" name="dob"></td>
</tr>

<!-- Gender (Radio buttons) -->
<tr>
<td>Gender:</td>
<td>
<input type="radio" name="gender" value="Male" id="male"><label
for="male">Male</label>
<input type="radio" name="gender" value="Female" id="female"><label
for="female">Female</label>

```

```

<input type="radio" name="gender" value="Other" id="other"><label
for="other">Other</label>
</td>
</tr>



<!-- Hobbies (Checkboxes) -->
<tr>
<td>Hobbies:</td>
<td>
<input type="checkbox" name="hobby" value="Reading"> Reading
<input type="checkbox" name="hobby" value="Music"> Music
<input type="checkbox" name="hobby" value="Sports"> Sports
<input type="checkbox" name="hobby" value="Travel"> Travel
</td>
</tr>
<!-- Country (Dropdown/Select Box) -->
<tr>
<td><label for="country">Country:</label></td>
<td>
<select id="country" name="country">
<option value="">--Select Country--</option>
<option value="India">India</option>
<option value="USA">USA</option>
<option value="UK">UK</option>
<option value="Australia">Australia</option>
</select>
</td>
</tr>
<!-- Address (Textarea) -->
<tr>
<td><label for="address">Address:</label></td>
<td><textarea id="address" name="address" rows="4" cols="30"></textarea></td>
</tr>




<!-- Buttons -->
<tr>
<td colspan="2" align="center">
<input type="submit" value="Submit">
<input type="reset" value="Reset">
</td>
</tr>
</table>
</form>
</body>
</html>

```

Output:

User Registration Form

Registration Details	
Full Name:	<input type="text"/>
Password:	<input type="password"/>
Age:	<input type="text"/> 
Date of Birth:	<input type="text" value="dd / mm / yyyy"/> 
Gender:	<input type="radio"/> Male <input type="radio"/> Female <input type="radio"/> Other
Hobbies:	<input type="checkbox"/> Reading <input type="checkbox"/> Music <input type="checkbox"/> Sports <input type="checkbox"/> Travel
Country:	<input type="text" value="--Select Country--"/> 
Address:	<div><input type="text"/></div> 
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

2d. 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).

Program:

Image

```
<!DOCTYPE html>
<html>
<body>

</body>
```

</html>

Frame

```
<!DOCTYPE html>
<html>
<head>
<title>Frame Example</title>
</head>
```

```
<frameset cols="33%,34%,33%" noresize border="1">
<frame src="image.html" name="imageFrame" noresize>
<frame src="p.html" name="paragraphFrame" noresize>
<frame src="l.html" name="linkFrame" noresize>
```

```
<noframes>
<body>
<p>Your browser does not support frames. Please use a modern browser.</p>
</body>
</noframes>
</frameset>
</html>
```

Sample Image

This is a sample paragraph displayed in the second frame.

[Visit OpenAI](#)

```
<!DOCTYPE html>
<html>
<body>
<a href="https://www.openai.com" target="_blank">Visit OpenAI</a>
</body>
</html>
```

[Visit OpenAI](#)

```
<!DOCTYPE html>
<html>
<body>
<p>This is a sample paragraph displayed in the second frame.</p>
</body>
```

</html>

This is a sample paragraph displayed in the second frame.

3. HTML 5 and Cascading Style Sheets, Types of CSS

a. Write a HTML program, that makes use of <article>, <aside>, <figure>, <figcaption>, <footer>, <header>, <main>, <nav>, <section>, <div>, tags.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>HTML5 Semantic Elements Example</title>
<style>
    body {
        font-family: Arial, sans-serif;
    }
    header, nav, main, footer {
        padding: 10px;
        border: 1px solid #ccc;
        margin: 5px;
    }
    aside {
        background-color: #f9f9f9;
        padding: 10px;
    }
    figure {
        margin: 10px 0;
    }
    .highlight {
        color: red;
        font-weight: bold;
    }
</style>
</head>
<body>

<header>
```

```
<h1>My Personal Blog</h1>
<p>Welcome to my world of thoughts.</p>
</header>
```

```
<nav>
<ul>
<li><a href="#">Home</a></li>
<li><a href="#">Articles</a></li>
<li><a href="#">Gallery</a></li>
<li><a href="#">Contact</a></li>
</ul>
</nav>
```

```
<main>
<section>
<h2>Latest Articles</h2>
```

```
<article>
<h3>Why Learn HTML5?</h3>
<p>HTML5 provides semantic tags that make your code cleaner and more accessible.
    Here's an example of a <span class="highlight">highlighted</span> word
    using the &lt;span&gt; tag.
</p>
```

```
<figure>

<figcaption>An example image using the &lt;figure&gt; and &lt;figcaption&gt;
tags.</figcaption>
</figure>
</article>
</section>
```

```
<aside>
<h3>About the Author</h3>
<p>This blog is maintained by a passionate web developer sharing tips and
tutorials.</p>
</aside>
```

```
<section>
<div>
<h3>Tech News</h3>
```

Output:

My Personal Blog

Welcome to my world of thoughts.

- [Home](#)
- [Articles](#)
- [Gallery](#)
- [Contact](#)

Latest Articles

Why Learn HTML5?

HTML5 provides semantic tags that make your code cleaner and more accessible. Here's an example of a **highlighted** word using the `` tag.

Sample Image

An example image using the `<figure>` and `<figcaption>` tags.

About the Author

This blog is maintained by a passionate web developer sharing tips and tutorials.

Tech News

3b. Write a HTML program, to embed audio and video into HTML web page.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Embedded Audio and Video Example</title>
</head>
<body>
<h1>Media Embedding with Online Sources</h1>
<!-- Audio Section -->
<section>
<h2>Listen to Nature Sounds</h2>
<audio controls>
<source src="https://www.soundhelix.com/examples/mp3/SoundHelix-Song-1.mp3"
type="audio/mpeg">
    Your browser does not support the audio element.
</audio>
</section>

<hr>

<!-- Video Section -->
<section>
<h2>Watch Sample Video</h2>
<video width="500" height="300" controls>
```



```
<source src="http://commondatastorage.googleapis.com/gtv-videos-  
bucket/sample/BigBuckBunny.mp4" type="video/mp4">
```

Your browser does not support the video tag.

```
</video>  
</section>
```

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

Output:

Media Embedding with Online Sources

Listen to Nature Sounds



Watch Sample Video



3C. 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).

Program:

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="UTF-8">  
<title>CSS Style Levels Example</title>  
  
<!-- Internal CSS -->  
<style>
```

```

    h1 {
        color: blue;
        text-align: center;
    }

    #internal-text {
        font-style: italic;
    }
</style>

<!-- Link to External CSS -->
<link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>Different Types of CSS Styles</h1>

<!-- Inline CSS -->
<p style="color: red; font-weight: bold;">
    This paragraph uses <strong>inline CSS</strong>.<br>
    Selector: `p` | Property: color | Value: red
</p>

<!-- Internal CSS -->
<p id="internal-text">
    This paragraph uses <strong>internal CSS</strong> (styled by ID selector).
</p>

<!-- External CSS -->
<p class="external-style">
    This paragraph uses <strong>external CSS</strong> (styled by class selector).
</p>

</body>
</html>

```

Styles :

```

* styles.css - External CSS file */
body {
    background-color: #f0f0f0;
    font-family: Arial, sans-serif;
}

.external-style {
    color: green;
    font-size: 20px;
}

```

Output:

Different Types of CSS Styles

This paragraph uses **inline CSS**.
Selector: 'p' | Property: color | Value: red

This paragraph uses **internal CSS** (styled by ID selector).

This paragraph uses **external CSS** (styled by class selector).

4. Selector forms a. Write a program to apply different types of selector forms i. Simple selector (element, id, class, group, universal) ii. Combinator selector (descendant, child, adjacent sibling, general sibling) iii. Pseudo-class selector iv. Pseudo-element selector v. Attribute selector

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>CSS Selector Types Example</title>
<style>
  /* =====
    i. Simple Selectors
    ===== */
  /* Element selector */
  h1 {
    color: darkblue;
  }

  /* ID selector */
  #highlight {
    background-color: yellow;
  }

  /* Class selector */
  .important {
    color: red;
    font-weight: bold;
  }

  /* Group selector */
  h2, p {
```

```
font-family: Arial, sans-serif;
}
```

```
/* Universal selector */
* {
    box-sizing: border-box;
}
```

```
/* =====
    ii. Combinator Selectors
===== */
```

```
/* Descendant selector */
div p {
    color: green;
}
```

```
/* Child selector */
ul > li {
    color: blue;
}
```

```
/* Adjacent sibling selector */
h2 + p {
    font-style: italic;
}
```

```
/* General sibling selector */
h3 ~ p {
    border-left: 3px solid gray;
    padding-left: 5px;
}
```

```
/* =====
    iii. Pseudo-class Selector
===== */
```

```
a:hover {
    color: orange;
}
```

```
li:first-child {
    font-style: italic;
}
```

```
/* =====
    iv. Pseudo-element Selector
===== */
```

```
p::first-line {
```

```

    font-weight: bold;
}

p::after {
    content: " ";
}

/* =====
   v. Attribute Selector
   ===== */
input[type="text"] {
    border: 2px solid teal;
}

a[target="_blank"] {
    text-decoration: underline;
}
</style>
</head>
<body>

<h1>CSS Selector Types Demo</h1>

<h2>Simple Selectors</h2>
<p>This is a simple paragraph.</p>
<p id="highlight">This paragraph uses an ID selector.</p>
<p class="important">This paragraph uses a class selector.</p>

<h2>Combinator Selectors</h2>
<div>
<p>This is a descendant paragraph (inside a div).</p>
</div>
<ul>
<li>First item (child selector)</li>
<li>Second item</li>
</ul>
<h2>Title</h2>
<p>This paragraph is adjacent to an h2 (adjacent sibling).</p>
<h3>Section Heading</h3>
<p>This paragraph is a general sibling of h3.</p>

<h2>Pseudo-class Selector</h2>
<a href="https://example.com" target="_blank">Hover over me</a>
<ul>
<li>First</li>
<li>Second</li>
</ul>

```

<h2>Pseudo-element Selector</h2>

<p>This is an example of a pseudo-element applied to the first line and end of paragraph.</p>

<h2>Attribute Selector</h2>

<input type="text" placeholder="Text input">

<input type="password" placeholder="Password input">

</body>

</html>

Output:

CSS Selector Types Demo

Simple Selectors

This is a simple paragraph. ← END

This paragraph uses an ID selector. ← END

This paragraph uses a class selector. ← END

Combinator Selectors

This is a descendant paragraph (inside a div). ← END

- *First item (child selector)*
- Second item

Title

This paragraph is adjacent to an h2 (adjacent sibling). ← END

Section Heading

| This paragraph is a general sibling of h3. ← END

Pseudo-class Selector

Hover over me

- *First*
- Second

Pseudo-element Selector

| This is an example of a pseudo-element applied to the first line and end of paragraph. ← END

Attribute Selector

5. CSS with Color, Background, Font, Text and CSS Box Model a. Write a program to demonstrate the various ways you can reference a color in CSS.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>CSS Color Reference Types</title>
  <style>
    /* 1. Named Color */
    .named-color {
      background-color: red;
      color: white;
      padding: 10px;
    }

    /* 2. Hexadecimal Color */
    .hex-color {
      background-color: #3498db;
      color: white;
      padding: 10px;
    }

    /* 3. RGB Color */
    .rgb-color {
      background-color: rgb(60, 179, 113); /* medium sea green */
      color: white;
      padding: 10px;
    }

    /* 4. RGBA Color */
    .rgba-color {
      background-color: rgba(255, 99, 71, 0.6); /* tomato with transparency */
      color: black;
      padding: 10px;
    }

    /* 5. HSL Color */
    .hsl-color {
      background-color: hsl(120, 100%, 25%); /* dark green */
      color: white;
      padding: 10px;
    }
  </style>
</head>
</html>
```

```

/* 6. HSLA Color */
.hsla-color {
  background-color: hsla(240, 100%, 50%, 0.5); /* semi-transparent blue */
  color: black;
  padding: 10px;
}

/* Styling for layout */
div {
  margin: 10px 0;
  font-family: Arial, sans-serif;
}
</style>
</head>
<body>

<h1>Ways to Reference Color in CSS</h1>

<div class="named-color">Named Color: red</div>
<div class="hex-color">Hexadecimal Color: #3498db</div>
<div class="rgb-color">RGB Color: rgb(60, 179, 113)</div>
<div class="rgba-color">RGBA Color: rgba(255, 99, 71, 0.6)</div>
<div class="hsl-color">HSL Color: hsl(120, 100%, 25%)</div>
<div class="hsla-color">HSLA Color: hsla(240, 100%, 50%, 0.5)</div>

</body>
</html>

```

Output:

Ways to Reference Color in CSS

Named Color: red

Hexadecimal Color: #3498db

RGB Color: rgb(60, 179, 113)

RGBA Color: rgba(255, 99, 71, 0.6)

HSL Color: hsl(120, 100%, 25%)

HSLA Color: hsla(240, 100%, 50%, 0.5)

5b. 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.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Fixed & Flipped Background Image with Pseudo-element</title>
<style>
  /* Container styling */
  .background-container {
    position: relative;
    height: 2000px; /* Allow scrolling for demonstration */
    z-index: 1; /* Ensures the content is above the pseudo-element */
    padding: 50px;
    color: white;
    text-align: center;
    font-family: Arial, sans-serif;
  }

  /* Styling for the pseudo-element that holds the background image */
  .background-container::before {
    content: ""; /* Required for the pseudo-element */
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-image: url('https://www.w3schools.com/w3images/forest.jpg'); /*
Using the image link */
    background-repeat: no-repeat;
    background-position: center 50%; /* Image positioned halfway down */
    background-attachment: fixed; /* Image stays in place during scroll */
    background-size: cover; /* Ensure the image covers the entire container */
    transform: scaleX(-1); /* Flips the image horizontally */
    z-index: -1; /* Keeps the pseudo-element behind the content */
  }

  /* Example content styling */
  .content {
    z-index: 2;
  }
</style>
</head>
<body>

<div class="background-container">
<div class="content">
```

```
<h1>Fixed & Flipped Background Image with Pseudo-element</h1>
```

```
<p>This background image stays fixed and tilted horizontally, and it's applied using a pseudo-element.</p>
```

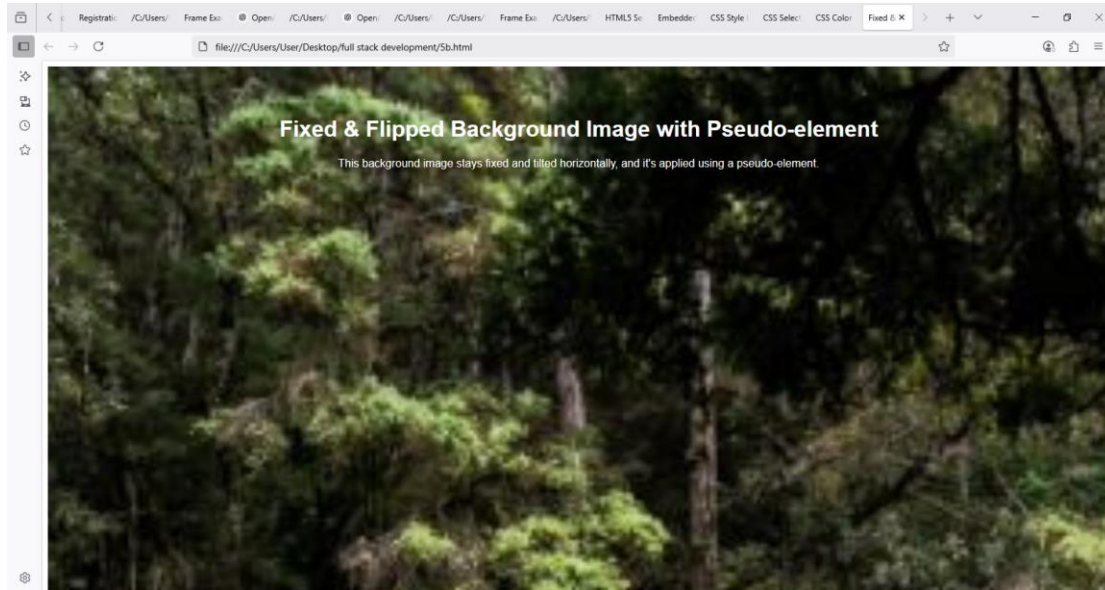
```
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```

Output:



5c. 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

Program:

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Font and Text Properties Example</title>
```

```
<style>
```

```
  body {
```

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

```
    padding: 30px;
```

```
  }
```

```
  .font-size-example {
```

```
    font-size: 24px;
```

```
  }
```

```
.font-weight-example {
  font-weight: bold;
}

.font-style-example {
  font-style: italic;
}

.text-decoration-example {
  text-decoration: underline;
}

.text-transform-example {
  text-transform: uppercase;
}

.text-align-example {
  text-align: center;
  background-color: #f0f0f0;
  padding: 10px;
}
</style>
</head>
<body>

<h1>CSS Font and Text Properties Demo</h1>

<p class="font-size-example">This text uses <strong>font-size: 24px</strong>.</p>

<p class="font-weight-example">This text uses <strong>font-weight:
bold</strong>.</p>

<p class="font-style-example">This text uses <strong>font-style: italic</strong>.</p>

<p class="text-decoration-example">This text uses <strong>text-decoration:
underline</strong>.</p>

<p class="text-transform-example">This text uses <strong>text-transform:
uppercase</strong>.</p>

<div class="text-align-example">
<p>This paragraph uses <strong>text-align: center</strong>.</p>
</div>

</body>
</html>
```

Output:

CSS Font and Text Properties Demo

This text uses **font-size: 24px**.

This text uses **font-weight: bold**.

This text uses **font-style: italic**.

This text uses **text-decoration: underline**.

THIS TEXT USES **TEXT-TRANSFORM: UPPERCASE**.

This paragraph uses **text-align: center**.

5d. . Write a program, to explain the importance of CSS Box model using i. Content ii. Border iii. Margin iv. padding

Program:

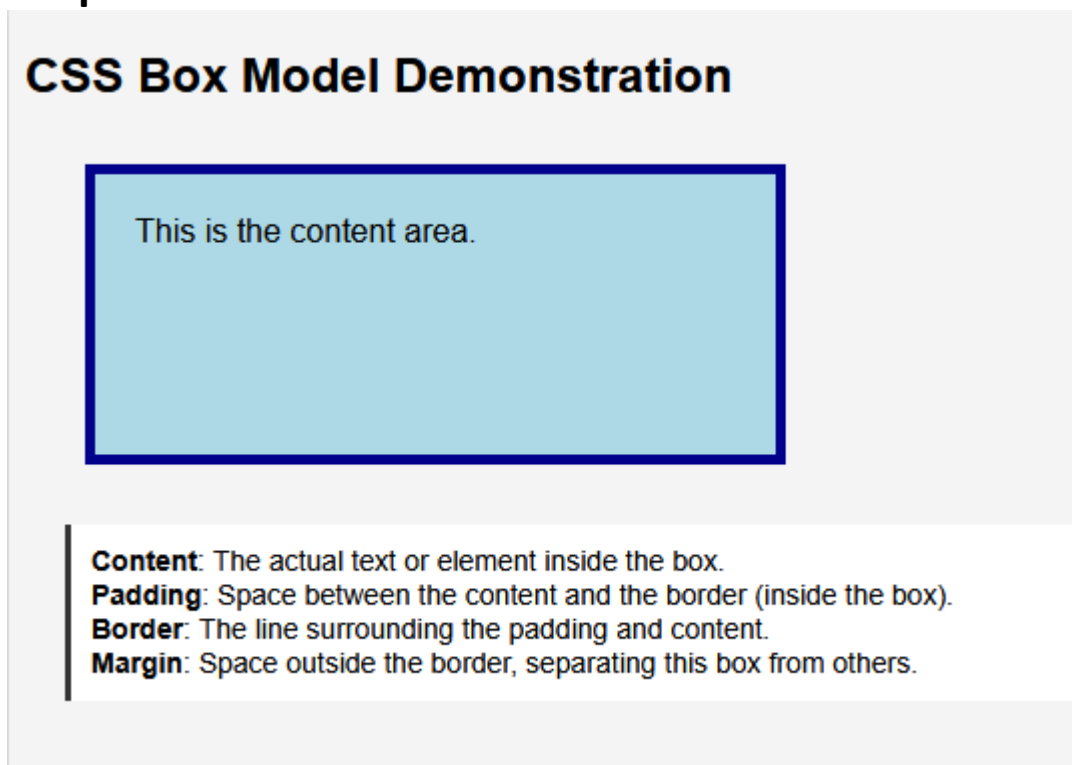
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>CSS Box Model Example</title>
<style>
  .box {
    width: 300px;
    height: 100px;
    background-color: lightblue;    /* Content area */
    padding: 20px;                 /* Space inside the box, around content */
    border: 5px solid darkblue;    /* Border surrounding padding */
    margin: 30px;                  /* Space outside the box */
  }

  body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
  }

  .info {
    font-size: 14px;
    background-color: #fff;
    padding: 10px;
    margin: 20px;
    border-left: 3px solid #333;
```

```
    }  
</style>  
</head>  
<body>  
  
<h2>CSS Box Model Demonstration</h2>  
  
<div class="box">  
    This is the content area.  
</div>  
  
<div class="info">  
<strong>Content</strong>: The actual text or element inside the box.<br>  
<strong>Padding</strong>: Space between the content and the border (inside the  
box).<br>  
<strong>Border</strong>: The line surrounding the padding and content.<br>  
<strong>Margin</strong>: Space outside the border, separating this box from others.  
</div>  
  
</body>  
</html>
```

Output:



- 6. Applying JavaScript - internal and external, I/O, Type Conversion**
- a. Write a program to embed internal and external JavaScript in a web page.**

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Internal and External JavaScript</title>

<!-- External JavaScript -->
<script>
function showExternalMessage() {
    alert("This is an External JavaScript alert!");
}
</script>

<!-- Internal JavaScript -->
<script>
    function showInternalMessage() {
        alert("This is an Internal JavaScript alert!");
    }
</script>
</head>
<body>

<h1>JavaScript Embedding</h1>

<button onclick="showInternalMessage()">Run Internal Script</button>
<button onclick="showExternalMessage()">Run External Script</button>

</body>
</html>
```

Output:

JavaScript Embedding

6b. Write a program to explain the different ways for displaying output.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
```

```

<title>JavaScript Output Methods</title>
</head>
<body>

<h2>JavaScript Output Examples</h2>

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

<script>
  // 1. Alert Box
  alert("This is an alert box (popup)");

  // 2. Document Write (use sparingly)
  document.write("This is written using document.write()<br>");

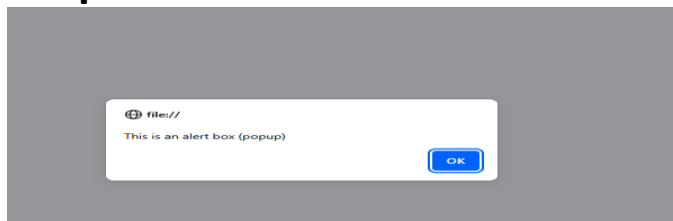
  // 3. Console Log (for developers)
  console.log("This message is logged to the browser console");

  // 4. InnerHTML (most common for updating page content)
  document.getElementById("output-area").innerHTML = "This content was inserted using
innerHTML!";
</script>

</body>
</html>

```

Output:



6C. Write a program to explain the different ways for taking input.

Program:

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>JavaScript Input Methods</title>
<style>
  body {
    font-family: Arial, sans-serif;
    padding: 20px;
  }
  input {
    padding: 5px;
    margin: 10px 0;
  }

```

```
</style>
</head>
<body>

<h2>JavaScript Input Methods</h2>

<!-- Method 1: Input using prompt() -->
<button onclick="promptInput()">Prompt Input</button>
<p id="prompt-result"></p>

<!-- Method 2: Input using HTML input field -->
<input type="text" id="textInput" placeholder="Enter your name">
<button onclick="readTextInput()">Submit</button>
<p id="input-result"></p>

<!-- Method 3: Input using confirm() -->
<button onclick="confirmInput()">Confirm Box</button>
<p id="confirm-result"></p>

<script>
  // Method 1: prompt()
  function promptInput() {
    let name = prompt("Enter your name:");
    if (name) {
      document.getElementById("prompt-result").innerText = "You entered: " + name;
    }
  }

  // Method 2: HTML input field
  function readTextInput() {
    let value = document.getElementById("textInput").value;
    document.getElementById("input-result").innerText = "Input from text field: " +
value;
  }

  // Method 3: confirm()
  function confirmInput() {
    let result = confirm("Do you like JavaScript?");
    document.getElementById("confirm-result").innerText = result
      ? "User clicked OK (Yes)"
      : "User clicked Cancel (No)";
  }
</script>

</body>
</html>
```


Output:

JavaScript Input Methods

Prompt Input

Enter your name

Submit

Confirm Box

6d. 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.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Voter Eligibility Check</title>
<style>
  body {
    font-family: Arial, sans-serif;
    padding: 20px;
  }
  table {
    width: 50%;
    border-collapse: collapse;
    margin-top: 20px;
  }
  th, td {
    padding: 10px;
    border: 1px solid #333;
    text-align: center;
  }
  th {
    background-color: #f2f2f2;
  }
}
```

```
</style>
</head>
<body>

<h2>Voter Information</h2>

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

<script>
  // Ask for user input
  let name = prompt("Enter your name:");
  let age = prompt("Enter your age:");

  // Convert age to number
  age = Number(age);

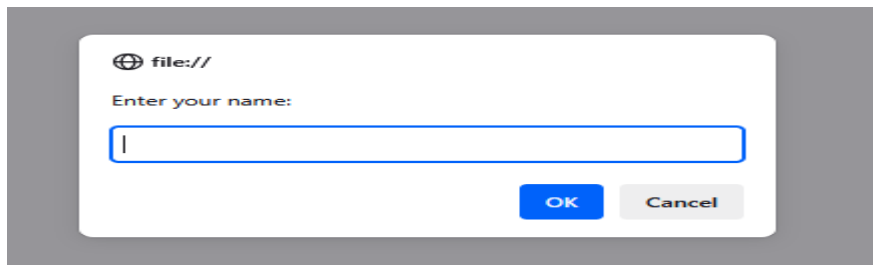
  // Determine voting eligibility
  let eligibility = (age >= 18) ? "Eligible to Vote" : "Not Eligible to Vote";

  // Display data in a table
  let tableHTML = `
<table>
<tr>
<th>Name</th>
<th>Age</th>
<th>Voting Eligibility</th>
</tr>
<tr>
<td>${name}</td>
<td>${age}</td>
<td>${eligibility}</td>
</tr>
</table>
`;

  // Output to the page
  document.getElementById("output").innerHTML = tableHTML;
</script>

</body>
</html>
```

Output:



7. JavaScript Pre-defined and User-defined Objects

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

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Document Object Example</title>
<script>
    function displayInfo() {
        // Accessing document properties
        let title = document.title;
        let url = document.URL;
        let lastModified = document.lastModified;

        // Displaying information in the page using document methods
        document.getElementById("info").innerHTML =
"<h3>Document Information</h3>" +
"<p><strong>Title:</strong>" + title + "</p>" +
"<p><strong>URL:</strong>" + url + "</p>" +
"<p><strong>Last Modified:</strong>" + lastModified + "</p>";
    }

    function changeTitle() {
        // Changing the document title using a document property
        document.title = "New Title Set by JavaScript";
        alert("Title changed to: " + document.title);
    }
</script>
</head>
<body onload="displayInfo()">
<h2>JavaScript Document Object Demo</h2>
<div id="info"></div>
<button onclick="changeTitle()">Change Document Title</button>
</body>
</html>
```

Output:

JavaScript Document Object Demo

Document Information

Title: Document Object Example

URL: file:///C:/Users/User/Desktop/full%20stack%20development/7a.html

Last Modified: 05/13/2025 13:58:46

Change Document Title

7B. Write a program using window object properties and methods.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Window Object Example</title>
<script>
    function showWindowProperties() {
        let width = window.innerWidth;
        let height = window.innerHeight;
        let locationHref = window.location.href;

        document.getElementById("output").innerHTML =
        "<h3>Window Properties</h3>" +
        "<p><strong>Width:</strong>" + width + " px</p>" +
        "<p><strong>Height:</strong>" + height + " px</p>" +
        "<p><strong>Current URL:</strong>" + locationHref + "</p>";
    }

    function openNewWindow() {
        // Opens a new browser window with specified dimensions
        window.open("https://www.example.com", "_blank",
        "width=600,height=400");
    }

    function confirmReload() {
        // Use confirm method of window
```

```

        if (window.confirm("Do you want to reload this page?")) {
            window.location.reload();
        }
    }

    function showAlert() {
        window.alert("This is a window alert!");
    }
</script>
</head>
<body onload="showWindowProperties()">
<h2>JavaScript Window Object Demo</h2>
<div id="output"></div>

<button onclick="openNewWindow()">Open New Window</button>
<button onclick="confirmReload()">Confirm and Reload</button>
<button onclick="showAlert()">Show Alert</button>
</body>
</html>

```

Output:

JavaScript Window Object Demo

Window Properties

Width: 1549 px

Height: 767 px

Current URL: file:///C:/Users/User/Desktop/full%20stack%20development/7b.html

7C. Write a program using array object properties and methods.

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Array Object Example</title>
<script>
    function arrayOperations() {
        // Declare an array (user-defined)
        let fruits = ["Apple", "Banana", "Mango"];

        // Using array methods and properties
        fruits.push("Orange"); // Adds an element to the end
    }

```

```

    fruits.unshift("Pineapple");    // Adds an element to the beginning
    let removedItem = fruits.pop();  // Removes the last element
    fruits.sort();                  // Sorts the array alphabetically
    let totalItems = fruits.length; // Array length

    // Display results
    document.getElementById("output").innerHTML =
"<h3>Array Operations</h3>" +
"<p><strong>Final Array:</strong>" + fruits.join(", ") + "</p>" +
"<p><strong>Removed Item:</strong>" + removedItem + "</p>" +
"<p><strong>Total Items:</strong>" + totalItems + "</p>";
  }
</script>
</head>
<body>
<h2>JavaScript Array Object Demo</h2>
<button onclick="arrayOperations()">Perform Array Operations</button>
<div id="output"></div>
</body>
</html>

```

Output:

JavaScript Array Object Demo

Perform Array Operations

7D. Write a program using math object properties and methods.

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Math Object Example</title>
<script>
  function calculateMathValues() {
    // Generating a random number between 0 and 1
    let randomNum = Math.random();

    // Using Math methods and properties
    let piValue = Math.PI;
    let rounded = Math.round(randomNum * 100);
    let squareRoot = Math.sqrt(64);
    let power = Math.pow(2, 4); // 2 to the power of 4
    let maxVal = Math.max(5, 10, 15, 20);
    let minVal = Math.min(5, 10, 15, 20);
  }

```

```

        // Display the results
        document.getElementById("output").innerHTML =
        "<h3>Math Object Operations</h3>" +
        "<p><strong>Random Number (0-1):</strong>" + randomNum.toFixed(4) + "</p>" +
        "<p><strong>Rounded (×100):</strong>" + rounded + "</p>" +
        "<p><strong>Value of PI:</strong>" + piValue + "</p>" +
        "<p><strong>Square Root of 64:</strong>" + squareRoot + "</p>" +
        "<p><strong>2<sup>4</sup>:</strong>" + power + "</p>" +
        "<p><strong>Max of [5, 10, 15, 20]:</strong>" + maxVal + "</p>" +
        "<p><strong>Min of [5, 10, 15, 20]:</strong>" + minVal + "</p>";
    }
</script>
</head>
<body>
<h2>JavaScript Math Object Demo</h2>
<button onclick="calculateMathValues()">Calculate Math Values</button>
<div id="output"></div>
</body>
</html>

```

Output:

JavaScript Math Object Demo

Calculate Math Values

7E. Write a program using string object properties and methods.

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>String Object Example</title>
<script>
    function stringOperations() {
        // Create a string
        let message = " Hello, JavaScript World! ";

        // Use string methods and properties
    }

```

```

    let length = message.length;
    let trimmed = message.trim();      // Remove whitespace
    let upper = trimmed.toUpperCase();  // Convert to uppercase
    let lower = trimmed.toLowerCase();  // Convert to lowercase
    let sliced = trimmed.slice(7, 17);  // Extract part of the string
    let replaced = trimmed.replace("JavaScript", "String");
    let includesCheck = trimmed.includes("World");

    // Display results
    document.getElementById("output").innerHTML =
"<h3>String Operations</h3>" +
"<p><strong>Original:</strong> " + message + "</p>" +
"<p><strong>Trimmed:</strong> " + trimmed + "</p>" +
"<p><strong>Length:</strong> " + length + "</p>" +
"<p><strong>Uppercase:</strong> " + upper + "</p>" +
"<p><strong>Lowercase:</strong> " + lower + "</p>" +
"<p><strong>Sliced (7-17):</strong> " + sliced + "</p>" +
"<p><strong>Replaced 'JavaScript' with 'String':</strong> " + replaced + "</p>" +
"<p><strong>Includes 'World':</strong> " + includesCheck + "</p>";
  }
</script>
</head>
<body>
<h2>JavaScript String Object Demo</h2>
<button onclick="stringOperations()">Perform String Operations</button>
<div id="output"></div>
</body>
</html>

```

Output:

JavaScript String Object Demo

Perform String Operations

7F. Write a program using regex object properties and methods.

Program:

```

<!DOCTYPE html>
<html>
<head>

```



```

<title>RegExp Object Example</title>
<script>
    function validateInput() {
        let userInput = document.getElementById("textInput").value;

        // Define a regular expression pattern (e.g., for a simple email)
        let regex = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}$/;

        // Test the pattern against the input
        let isValid = regex.test(userInput);

        // Display the result
        let resultMessage = isValid
            ? "✔Valid email address!"
            : "✘Invalid email address format.";

        document.getElementById("output").innerHTML =
            "<p><strong>Input:</strong>" + userInput + "</p>" +
            "<p><strong>Result:</strong>" + resultMessage + "</p>" +
            "<p><strong>Pattern Used:</strong>" + regex.toString() + "</p>";
    }
</script>
</head>
<body>
<h2>JavaScript RegExp Object Demo</h2>
<label for="textInput">Enter Email Address:</label>
<input type="text" id="textInput">
<button onclick="validateInput()">Validate</button>
<div id="output"></div>
</body>
</html>

```

Output:

JavaScript RegExp Object Demo

Enter Email Address:

7G. Write a program using date object properties and methods.

Program:

```
<!DOCTYPE html>
```

```

<html>
<head>
<title>Date Object Example</title>
<script>
    function showDateInfo() {
        // Create a new Date object for the current date and time
        let now = new Date();

        // Extract date and time details using Date methods
        let dateString = now.toString(); // e.g., "Tue May 13 2025"
        let timeString = now.toTimeString(); // e.g., "13:45:30 GMT+0000"
        let year = now.getFullYear(); // e.g., 2025
        let month = now.getMonth() + 1; // Months are zero-based (0 = Jan)
        let day = now.getDate(); // Day of the month
        let weekday = now.getDay(); // Day of the week (0 = Sunday)

        const days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"];

        // Display the results
        document.getElementById("output").innerHTML =
"<h3>Date Object Information</h3>" +
"<p><strong>Full Date:</strong>" + dateString + "</p>" +
"<p><strong>Time:</strong>" + timeString + "</p>" +
"<p><strong>Year:</strong>" + year + "</p>" +
"<p><strong>Month:</strong>" + month + "</p>" +
"<p><strong>Day of Month:</strong>" + day + "</p>" +
"<p><strong>Day of Week:</strong>" + days[weekday] + "</p>";
    }
</script>
</head>
<body>
<h2>JavaScript Date Object Demo</h2>
<button onclick="showDateInfo()">Show Current Date & Time</button>
<div id="output"></div>
</body>
</html>

```

Output:

JavaScript Date Object Demo

Show Current Date & Time

7H. Write a program to explain user-defined object by using properties, methods, accessors, constructors and display.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>User-defined Object Example</title>
<script>
    // Constructor function for a Student object
    function Student(name, age, grade) {
        // Properties
        this.name = name;
        this.age = age;
        this.grade = grade;

        // Method
        this.displayInfo = function() {
            return `Name: ${this.name}, Age: ${this.age}, Grade: ${this.grade}`;
        };

        // Accessor: Getter for grade
        this.getGrade = function() {
            return this.grade;
        };

        // Accessor: Setter for grade
        this.setGrade = function(newGrade) {
            this.grade = newGrade;
        };
    }

    function createAndDisplayStudent() {
        // Create a new student object using the constructor
        let student1 = new Student("Alice", 16, "A");

        // Use setter to change the grade
        student1.setGrade("A+");

        // Display student info
        document.getElementById("output").innerHTML =
        "<h3>Student Information</h3>" +
        "<p><strong>Using Method:</strong>" + student1.displayInfo() + "</p>" +
        "<p><strong>Using Getter:</strong> Grade is " + student1.getGrade() + "</p>";
    }
</script>
```

```
</head>
<body>
<h2>User-defined Object in JavaScript</h2>
<button onclick="createAndDisplayStudent()">Create and Display Student</button>
<div id="output"></div>
</body>
</html>
```

Output:

User-defined Object in JavaScript

Create and Display Student

8. JavaScript Conditional Statements and Loops

- a. 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”.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Compare Numbers</title>
</head>
<body>

<script>
  // Ask user for three integers
  let num1 = parseInt(prompt("Enter the first integer:"));
  let num2 = parseInt(prompt("Enter the second integer:"));
  let num3 = parseInt(prompt("Enter the third integer:"));

  // Check if all numbers are equal
  if (num1 === num2 && num2 === num3) {
    document.write("<p><strong>EQUAL NUMBERS</strong></p>");
  } else {
    // Find the largest number
    let largest = num1;

    if (num2 > largest) {
      largest = num2;
    }
  }
</script>
</body>
</html>
```

```

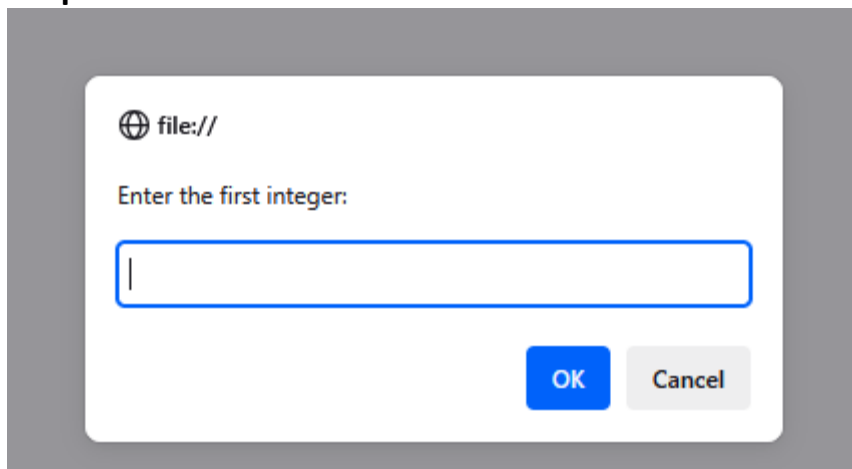
    if (num3 > largest) {
        largest = num3;
    }

    document.write(`<p><strong>${largest} LARGER NUMBER</strong></p>`);
}
</script>

</body>
</html>

```

Output:



8B. Write a program to display week days using switch case

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Week Days with Switch Case</title>
</head>
<body>

<script>
    // Ask the user to enter a number between 1 and 7
    let dayNumber = parseInt(prompt("Enter a number (1-7) to get the day of the
week:"));

    // Use switch case to determine the day
    switch (dayNumber) {
        case 1:
            document.write("<p>Monday</p>");
            break;
        case 2:
            document.write("<p>Tuesday</p>");

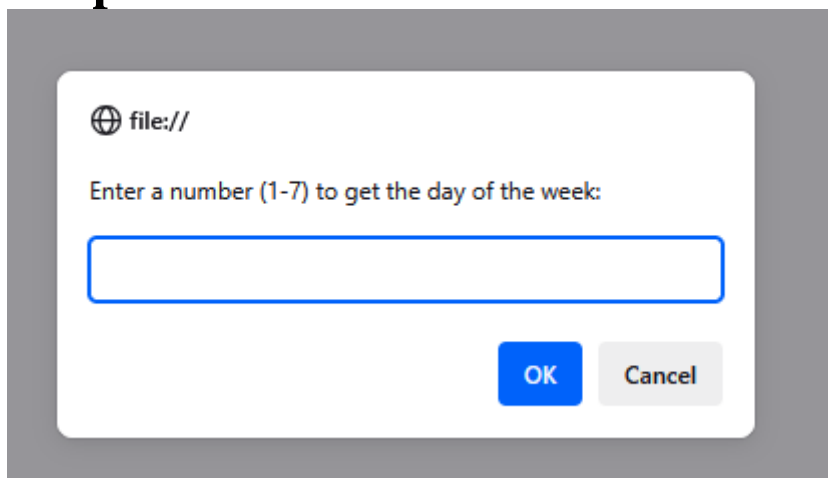
```

```

        break;
    case 3:
        document.write("<p>Wednesday</p>");
        break;
    case 4:
        document.write("<p>Thursday</p>");
        break;
    case 5:
        document.write("<p>Friday</p>");
        break;
    case 6:
        document.write("<p>Saturday</p>");
        break;
    case 7:
        document.write("<p>Sunday</p>");
        break;
    default:
        document.write("<p>Invalid input! Please enter a number between 1 and 7.</p>");
    }
</script>
</body>
</html>

```

Output:



8C. Write a program to print 1 to 10 numbers using for, while and do-while loops.

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Print 1 to 10 Using Loops</title>
</head>
<body>

```

<h2>Using for loop:</h2>

```
<script>
  for (let i = 1; i <= 10; i++) {
    document.write(i + "");
  }
</script>
```

<h2>Using while loop:</h2>

```
<script>
  let j = 1;
  while (j <= 10) {
    document.write(j + "");
    j++;
  }
</script>
```

<h2>Using do-while loop:</h2>

```
<script>
  let k = 1;
  do {
    document.write(k + "");
    k++;
  } while (k <= 10);
</script>
```

</body>

</html>

Output:

Using for loop:

1 2 3 4 5 6 7 8 9 10

Using while loop:

1 2 3 4 5 6 7 8 9 10

Using do-while loop:

1 2 3 4 5 6 7 8 9 10

8D. Write a program to print data in object using for-in, for-each and for-of loops.

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Loop Through Object Data</title>
</head>
<body>

<script>
  // Define an object
  const person = {
    name: "Alice",
    age: 30,
    city: "New York"
  };

  document.write("<h3>Using for...in (for object properties):</h3>");
  for (let key in person) {
    document.write(`${key}: ${person[key]}<br>`);
  }

  document.write("<h3>Using forEach (on Object.entries):</h3>");
  Object.entries(person).forEach(([key, value]) => {
    document.write(`${key}: ${value}<br>`);
  });

  document.write("<h3>Using for...of (on Object.entries):</h3>");
  for (let [key, value] of Object.entries(person)) {
    document.write(`${key}: ${value}<br>`);
  }
</script>

</body>
</html>
```

Output:

Using for...in (for object properties):

name: Alice
age: 30
city: New York

Using forEach (on Object.entries):

name: Alice
age: 30
city: New York

Using for...of (on Object.entries):

name: Alice
age: 30
city: New York

8E. 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., $1^3 + 5^3 + 3^3 = 153$]

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Armstrong Number Checker</title>
</head>
<body>

<script>
  // Ask the user for a number
  let number = parseInt(prompt("Enter a number to check if it's an Armstrong
number:"));

  // Convert number to string to work with digits
  let digits = number.toString();
  let numDigits = digits.length;
  let sum = 0;

  // Calculate sum of digits raised to the power of number of digits
  for (let i = 0; i < numDigits; i++) {
    sum += Math.pow(parseInt(digits[i]), numDigits);
  }
```

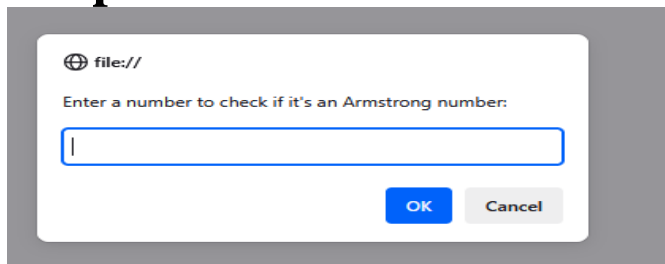
```

// Compare sum with original number
if (sum === number) {
    document.write(`<p><strong>${number} is an Armstrong number.</strong></p>`);
} else {
    document.write(`<p><strong>${number} is NOT an Armstrong
number.</strong></p>`);
}
</script>

</body>
</html>

```

Output:



8F. 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)

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Bank Denomination Calculator</title>
</head>
<body>

<script>
// Get the deposited amount from the user
let amount = parseInt(prompt("Enter the amount deposited (in Rs):"));
let originalAmount = amount;

// Denominations
const denominations = [100, 50, 20, 10, 5, 2, 1];

document.write(`<h3>Breakdown of Rs.${originalAmount}</h3>`);

denominations.forEach(denom => {
    let count = Math.floor(amount / denom);
    if (count > 0) {

```

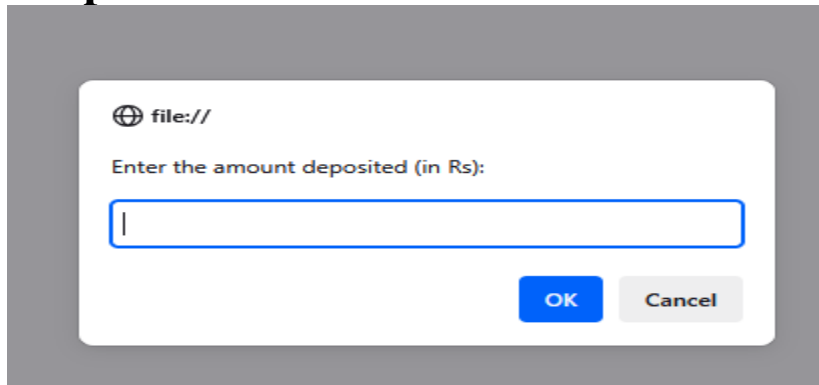
```

        document.write(`${count} x Rs.${denom}<br>`);
        amount = amount % denom;
    }
});
</script>

</body>
</html>

```

Output:



9. Java script Functions and Events

a. Design a appropriate function should be called to display

- i. Factorial of that number
- ii. Fibonacci series up to that number
- iii. Prime numbers up to that number
- iv. Is it palindrome or not

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Auto Number Analyzer</title>
<style>
    body {
        font-family: Arial;
        margin: 20px;
    }
    input {
        padding: 8px;
        width: 200px;
    }
    #output {
        margin-top: 20px;
    }
    .result {
        margin-bottom: 10px;
    }

```

```

</style>
</head>
<body>

<h2>Type a Number:</h2>
<input type="number" id="numberInput" placeholder="Enter a number">
<div id="output"></div>

<script>
const input = document.getElementById("numberInput");
const output = document.getElementById("output");

input.addEventListener("input", function () {
  const num = parseInt(input.value);
  output.innerHTML = ""; // Clear previous output

  if (isNaN(num) || num < 0) {
    output.innerHTML = "<div class='result'>Please enter a valid non-negative
number.</div>";
    return;
  }

  // Factorial
  let fact = 1;
  for (let i = 2; i <= num; i++) {
    fact *= i;
  }
  output.innerHTML += `<div class='result'>Factorial of ${num} is ${fact}</div>`;

  // Fibonacci Series
  let fib = [0, 1];
  while (fib[fib.length - 1] + fib[fib.length - 2] <= num) {
    fib.push(fib[fib.length - 1] + fib[fib.length - 2]);
  }
  const fibSeries = fib.filter(n => n <= num).join(", ");
  output.innerHTML += `<div class='result'>Fibonacci series up to ${num}:
${fibSeries}</div>`;

  // Prime Numbers
  const primes = [];
  for (let i = 2; i <= num; 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);
  }
}

```

```

    output.innerHTML += `<div class='result'>Prime numbers up to ${num}:
    ${primes.join(", ")}</div>`;

    // Palindrome Check
    const str = num.toString();
    const reversed = str.split("").reverse().join("");
    const isPalindrome = (str === reversed);
    output.innerHTML += `<div class='result'>${num} is ${isPalindrome ? "": "not "}a
    palindrome</div>`;
  });
</script>

</body>
</html>

```

Output:

Type a Number:

9B .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

- i. Factorial of that number
- ii. Fibonacci series up to that number
- iii. Prime numbers up to that number
- iv. Is it palindrome or not

Program:

```

<!DOCTYPE html>
<html>
<head>
<title>Number Analyzer</title>
<style>
  body {
    font-family: Arial;
    margin: 20px;
  }
  input, button {
    margin: 5px;
    padding: 8px;
  }

```

```

    #output {
        margin-top: 20px;
        font-weight: bold;
    }
</style>
</head>
<body>

<h2>Enter a Number:</h2>
<input type="number" id="numberInput" placeholder="Enter a number">

<br>

<button onclick="calculateFactorial()">Factorial</button>
<button onclick="generateFibonacci()">Fibonacci</button>
<button onclick="listPrimes()">Prime</button>
<button onclick="checkPalindrome()">Palindrome</button>

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

<script>
    function getInputNumber() {
        return parseInt(document.getElementById("numberInput").value);
    }

    function calculateFactorial() {
        const num = getInputNumber();
        if (isNaN(num) || num < 0) {
            document.getElementById("output").innerHTML = "Please enter a non-negative
integer.";
            return;
        }
        let fact = 1;
        for (let i = 2; i <= num; i++) {
            fact *= i;
        }
        document.getElementById("output").innerHTML = `Factorial of ${num} is
${fact}`;
    }

    function generateFibonacci() {
        const num = getInputNumber();
        if (isNaN(num) || num < 0) {
            document.getElementById("output").innerHTML = "Please enter a non-negative
number.";
            return;
        }

        const fib = [0, 1];
        while (fib[fib.length - 1] + fib[fib.length - 2] <= num) {

```

```

        fib.push(fib[fib.length - 1] + fib[fib.length - 2]);
    }

    const result = fib.filter(n => n <= num);
    document.getElementById("output").innerHTML = `Fibonacci series up to ${num}:
    ${result.join(', ')}`;
}

function listPrimes() {
    const num = getInputNumber();
    if (isNaN(num) || num < 2) {
        document.getElementById("output").innerHTML = "Enter a number greater than
1.";
        return;
    }

    const primes = [];
    for (let i = 2; i <= num; 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);
    }

    document.getElementById("output").innerHTML = `Prime numbers up to ${num}:
    ${primes.join(', ')}`;
}

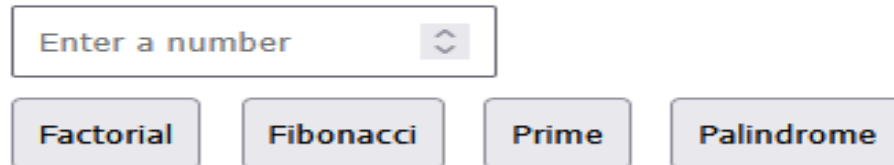
function checkPalindrome() {
    const num = getInputNumber();
    if (isNaN(num)) {
        document.getElementById("output").innerHTML = "Please enter a valid number.";
        return;
    }

    const str = num.toString();
    const reversed = str.split("").reverse().join("");
    if (str === reversed) {
        document.getElementById("output").innerHTML = `${num} is a palindrome.`;
    } else {
        document.getElementById("output").innerHTML = `${num} is not a palindrome.`;
    }
}
</script>
</body>
</html>

```

Output:

Enter a Number:



9C. Write a program to validate the following fields in a registration page.

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

Program:

```
<!DOCTYPE html>
<html>
<head>
<title>Registration Form Validation</title>
<style>
  body { font-family: Arial; padding: 20px; }
  input { margin-bottom: 10px; padding: 8px; width: 300px; }
  .error { color: red; font-size: 14px; }
  .success { color: green; font-weight: bold; }
</style>
</head>
<body>

<h2>Registration Form</h2>

<form id="registrationForm" onsubmit="return validateForm()">
<label>Name:</label><br>
<input type="text" id="name"><br>
<span id="nameError" class="error"></span><br>

<label>Mobile:</label><br>
<input type="text" id="mobile"><br>
<span id="mobileError" class="error"></span><br>

<label>Email:</label><br>
```



```
<input type="text" id="email"><br>
<span id="emailError" class="error"></span><br>

<input type="submit" value="Register">
</form>

<div id="successMsg" class="success"></div>

<script>
function validateForm() {
    // Clear previous messages
    document.getElementById("nameError").innerText = "";
    document.getElementById("mobileError").innerText = "";
    document.getElementById("emailError").innerText = "";
    document.getElementById("successMsg").innerText = "";

    let isValid = true;

    // Get values
    const name = document.getElementById("name").value.trim();
    const mobile = document.getElementById("mobile").value.trim();
    const email = document.getElementById("email").value.trim();

    // Name validation: starts with letter, followed by alphanumerics, min 6 characters
    const namePattern = /^[A-Za-z][A-Za-z0-9]{5,}$/;
    if (!namePattern.test(name)) {
        document.getElementById("nameError").innerText = "Invalid name. Must start
with a letter and be at least 6 characters.";
        isValid = false;
    }

    // Mobile validation: exactly 10 digits
    const mobilePattern = /^[0-9]{10}$/;
    if (!mobilePattern.test(mobile)) {
        document.getElementById("mobileError").innerText = "Mobile must be exactly
10 digits.";
        isValid = false;
    }

    // Email validation: basic format check
    const emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;
    if (!emailPattern.test(email)) {
        document.getElementById("emailError").innerText = "Invalid email format.";
        isValid = false;
    }

    if (isValid) {
        document.getElementById("successMsg").innerText = "Registration successful!";
    }
}
```

```
        return false; // prevent form submission for demo purposes
    }
</script>

</body>
</html>
```

Registration Form

Name:

Mobile:

Email:

Register