Name: Lao Rana

Class: A3

Subject: WCT

Homework Day5

**Task 1:**

**CSS (Cascading Style Sheets)** is a styling language used to define the appearance and layout of HTML elements on a webpage. CSS allows developers to control colors, fonts, spacing, layout, and other visual aspects of the HTML elements.

body {

    background-color: #f0f8ff; /\* Sets a light blue background \*/

    color: #333; /\* Sets text color to dark gray \*/

    font-size: 16px; /\* Sets default font size \*/

  }

  h1 {

    font-size: 2em; /\* Larger font size for header \*/

    color: #2c3e50; /\* Darker color for the header \*/

  }

  p {

    font-size: 1em;

    color: #555; /\* Sets paragraph text color \*/

  }

**Task 2:**

**Differences**:

* **Inline CSS**: Directly applied to individual elements via the style attribute.
* **Internal CSS**: is useful for one-page styles but less modular. Written within a <style> tag in the HTML <head>.
* **External CSS**: Linked from an external .css file, is the best for modular, reusable styles.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS Methods</title>

<style>

/\* Internal CSS \*/

body {

background-color: pink;

}

</style>

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

</head>

<body>

<h1 style="color: blue;">This is Inline CSS</h1> <!-- Inline CSS -->

<p>This paragraph is styled using external CSS.</p>

</body>

</html>

**Task 3:**

**CSS selectors** are patterns used to select and style HTML elements.

<!DOCTYPE html>

<html>

<head>

<style>

/\* Element selector \*/

p {

color: blue;

}

/\* Class selector \*/

.highlight {

color: orange;

}

/\* ID selector \*/

#unique {

font-weight: bold;

}

/\* Attribute selector \*/

a[target="\_blank"] {

color: red;

}

/\* Pseudo-class selector \*/

a:hover {

color: green;

}

</style>

</head>

<body>

<p>Normal paragraph.</p>

<p class="highlight">Highlighted paragraph.</p>

<p id="unique">Unique paragraph.</p>

<a href="#">Normal link</a>

<a href="#" target="\_blank">External link</a>

</body>

</html>

**Task 4:**

Debugging with Developer Tools:

1. Right-click an element > Inspect.
2. View styles in the "Elements" panel.
3. Adjust properties live and review results.

**Task 5:**

Using the style attribute directly in HTML applies inline styling.

<!DOCTYPE html>

<html>

<body>

<h1 style="color: green; font-size: 24px;">Inline Styled Header</h1>

<p style="color: gray;">This is an inline-styled paragraph.</p>

</body>

</html>

**Task 6:**

**Common CSS Properties for Text and Fonts**

1. font-family: Sets the font of the text. You can specify a list of fonts to fall back on if the first choice isn’t available.
2. **font-size**: Sets the size of the font. You can use units like px (pixels), em (relative to the parent element’s font size), rem (relative to the root element’s font size), and %.
3. **font-weight**: Sets the weight (thickness) of the text. Common values are normal, bold, and numbers like 300, 400, 700 (depending on the font family).

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Text and Font Styling</title>

<style>

/\* Body text \*/

body {

font-family: Arial, sans-serif;

color: #333;

line-height: 1.6;

}

/\* Headings \*/

h1 {

font-size: 2.5em;

font-weight: bold;

text-align: center;

text-transform: uppercase;

color: #2c3e50;

}

h2 {

font-size: 1.8em;

font-weight: 300;

color: #34495e;

text-decoration: underline;

}

/\* Paragraphs \*/

p {

font-size: 1em;

line-height: 1.5;

letter-spacing: 0.5px;

}

/\* Links \*/

a {

color: #3498db;

text-decoration: none;

}

a:hover {

text-decoration: underline;

}

/\* Emphasized text \*/

em {

font-style: italic;

color: #e74c3c;

}

</style>

</head>

<body>

<h1>Text and Font Styling Example</h1>

<h2>Subtitle Example</h2>

<p>This is an example paragraph showing how to style text and fonts in CSS. You can control the <strong>size</strong>, <em>color</em>, and spacing of the text to create a more readable and visually appealing layout.</p>

<p>Here is a link with <a href="#">hover effect</a> to demonstrate text-decoration and pseudo-classes.</p>

</body>

</html>

**Task 7:**

**Online fonts** are fonts that are hosted on the internet and can be accessed by your website through an external link, rather than being installed on the user’s device. These fonts are often provided by third-party services like Google Fonts, Adobe Fonts, or other font hosting platforms. By using online fonts, you can style your website with unique typography without needing to worry about whether the user has the specific font installed locally.

**Integrating Google Fonts into a Website**

Google Fonts is a popular and free service that provides access to a wide variety of fonts. You can easily integrate Google Fonts into your HTML document by including a link to the font file in the <head> section of your HTML. Here’s how to use Google Fonts in a web page:

1. **Visit Google Fonts** at <https://fonts.google.com> and browse for a font you like.
2. Select the font and choose specific styles (e.g., regular, bold).
3. Google Fonts will generate a <link> tag that you can copy and paste into your HTML document.

**Example**: Adding Google Fonts to Your HTML Document

/\* Google Fonts \*/

@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&display=swap");

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

  scroll-padding-top: 2rem;

  scroll-behavior: smooth;

  list-style: none;

  text-decoration: none;

  font-family: "Poppins", sans-serif;

}

**Task 8:**

In CSS, colors and backgrounds can be applied using several methods and formats. CSS supports color formats such as HEX, RGB, and HSL, and background properties like solid colors, gradients, and images.

/\* General styling \*/

body {

font-family: Arial, sans-serif;

display: flex;

flex-direction: column;

align-items: center;

gap: 20px;

margin: 20px;

}

/\* 1. Solid Background Color \*/

.solid-color {

background-color: #4CAF50; /\* HEX Color \*/

color: #FFFFFF; /\* Text color in HEX \*/

padding: 20px;

width: 300px;

text-align: center;

border-radius: 8px;

}

/\* 2. HEX Color \*/

.hex-color {

color: #FF6347; /\* Tomato color \*/

padding: 20px;

width: 300px;

text-align: center;

border: 2px solid #FF6347;

border-radius: 8px;

}

/\* 3. RGB Color \*/

.rgb-color {

color: rgb(70, 130, 180); /\* SteelBlue color \*/

padding: 20px;

width: 300px;

text-align: center;

border: 2px solid rgb(70, 130, 180);

border-radius: 8px;

}

/\* 4. HSL Color \*/

.hsl-color {

color: hsl(200, 70%, 50%); /\* SkyBlue color \*/

padding: 20px;

width: 300px;

text-align: center;

border: 2px solid hsl(200, 70%, 50%);

border-radius: 8px;

}

/\* 5. Gradient Background \*/

.gradient-background {

background: linear-gradient(45deg, #ff7e5f, #feb47b); /\* Gradient from pink to peach \*/

color: #FFFFFF;

padding: 20px;

width: 300px;

text-align: center;

border-radius: 8px;

}

/\* 6. Image Background \*/

.image-background {

background: url('https://via.placeholder.com/300') no-repeat center center / cover;

color: #FFFFFF;

padding: 20px;

width: 300px;

text-align: center;

border-radius: 8px;

}

**Task 9:**

In CSS, aligning elements can be done in several ways, depending on the element's display property. Common display properties include block, inline, inline-block, and flex, each of which affects how elements are positioned and aligned within a page layout. Here’s a guide to using these properties, alongside text alignment and vertical alignment techniques.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>CSS Alignment and Display Properties</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h1>CSS Alignment Techniques</h1>

<section class="text-alignment">

<h2>Text Alignment</h2>

<div class="left-align">Left Aligned Text</div>

<div class="center-align">Center Aligned Text</div>

<div class="right-align">Right Aligned Text</div>

</section>

<section class="display-properties">

<h2>Display Properties</h2>

<div class="block-element">Block Element</div>

<span class="inline-element">Inline Element</span>

<span class="inline-element">Another Inline Element</span>

<div class="inline-block-element">Inline-Block Element</div>

<div class="inline-block-element">Another Inline-Block Element</div>

</section>

<section class="flex-alignment">

<h2>Flexbox Alignment</h2>

<div class="flex-container">

<div class="flex-item">Flex Item 1</div>

<div class="flex-item">Flex Item 2</div>

<div class="flex-item">Flex Item 3</div>

</div>

</section>

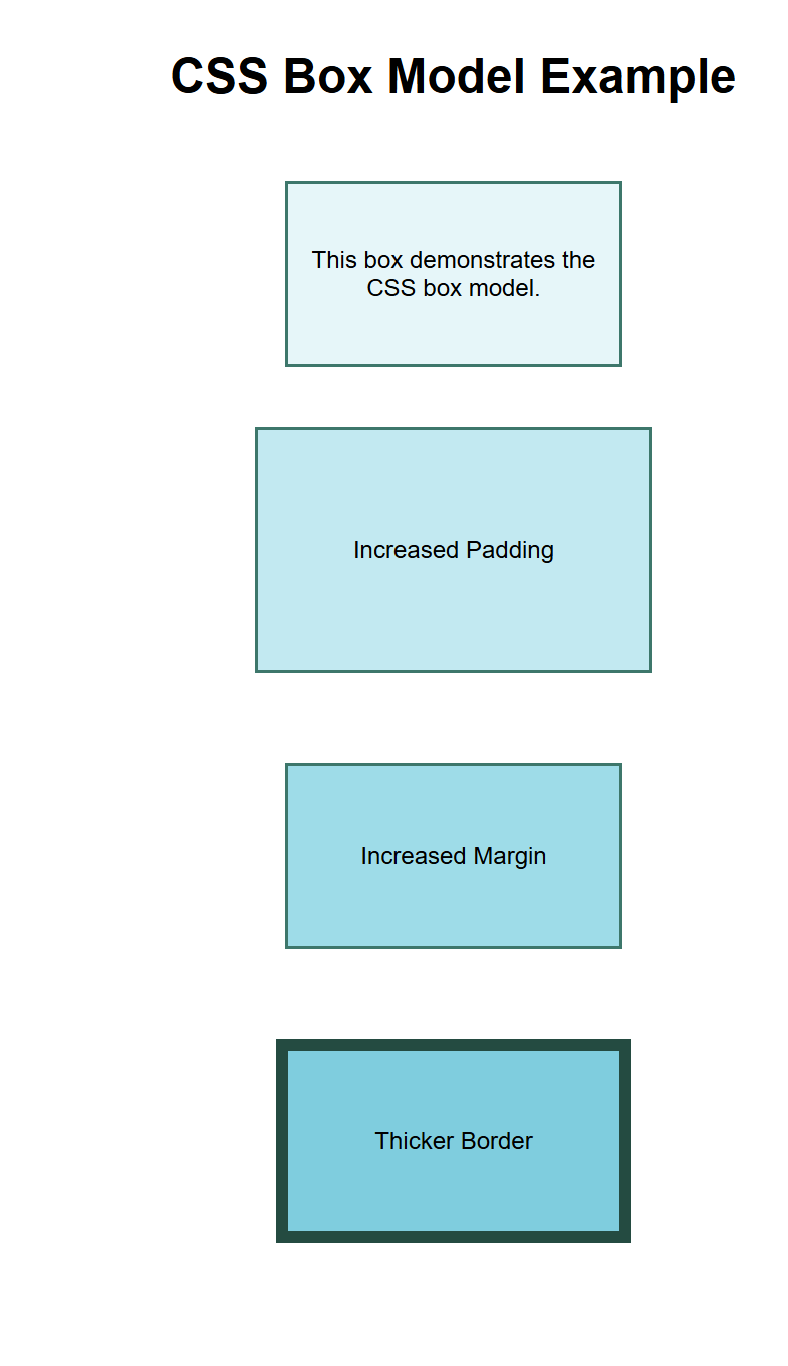
</body>

</html>

**Task 10:**

The CSS box model is a fundamental concept in web design, defining how elements are structured and spaced on a webpage. Each element on a page is treated as a rectangular box, which consists of the following components:

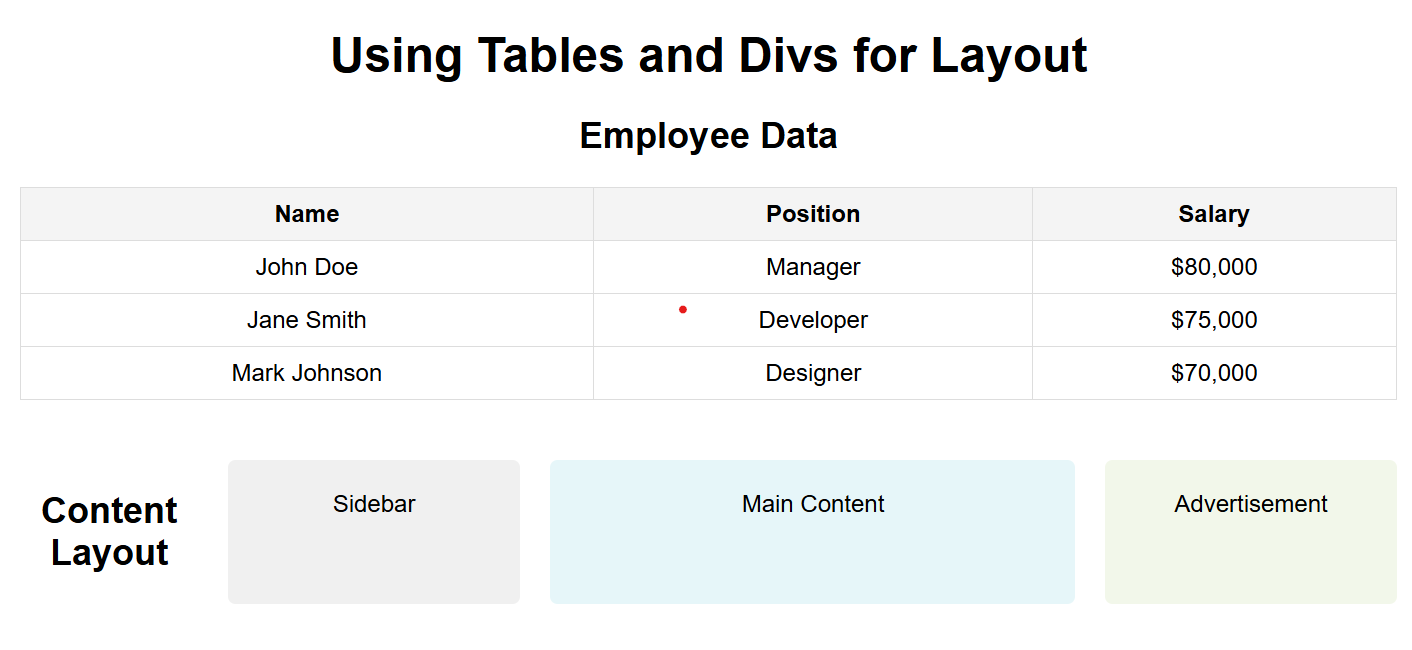
1. Content: The actual content of the box, such as text or images, defined by its width and height.
2. Padding: Space between the content and the border. Padding adds space inside the box, increasing the box’s size.
3. Border: A line surrounding the padding and content. The border can be customized with color, width, and style.
4. Margin: The space outside the border, creating space between the box and other elements.



**Task 11:**

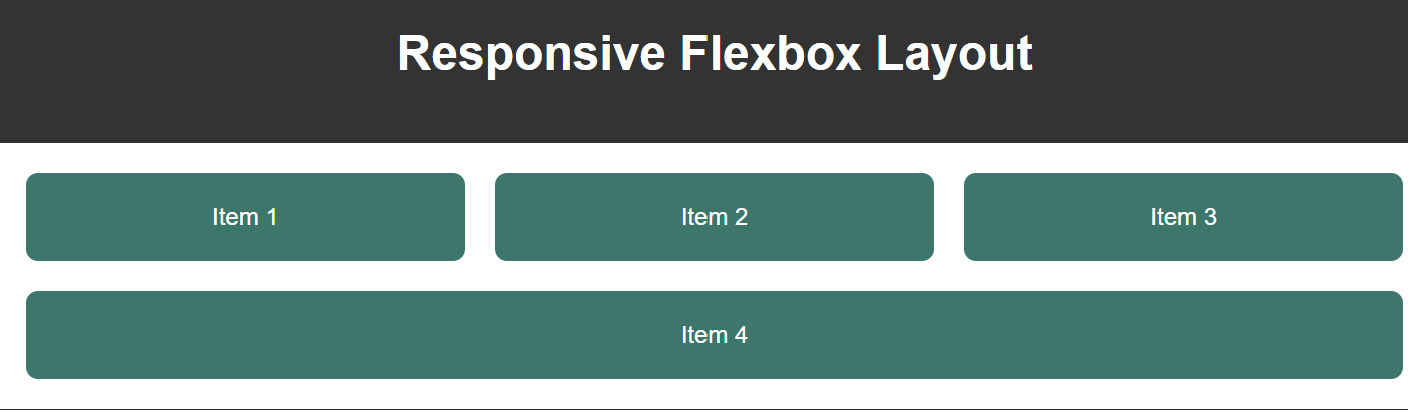
In web design, both <table> elements and <div> elements can be used for layout, but they serve different purposes and are used in different contexts. Here's a breakdown of when to use each and how to style them with CSS:

1. Tables:
   * Purpose: Tables are used to display tabular data (i.e., data arranged in rows and columns). They are semantically appropriate when presenting information like schedules, prices, or any dataset that naturally fits into a grid structure.
   * Layout: Tables should not be used for general page layout (such as positioning content, navigation, or sidebars) as they are meant for data presentation.
2. <div> Elements:
   * Purpose: <div> elements are used for grouping content and are the foundation for modern web layouts. They are versatile and commonly used in CSS-based layouts for positioning content, creating sections, grids, and flexible designs.
   * Layout: <div> elements allow for more flexible, responsive, and modern layouts compared to tables. They are often styled using CSS properties like display: flex, display: grid, or display: block to control the structure and positioning.



**Task 12:**

The Flexbox (Flexible Box Layout) model is a layout system in CSS that allows for more efficient and predictable layouts, especially when building responsive web designs. Flexbox provides a way to align and distribute space among items in a container, even when their size is unknown or dynamic. It simplifies the alignment of elements in both horizontal and vertical directions.



**Task 13:**

How CSS Variables Work

1. Defining a CSS Variable:
   * You define a CSS variable by prefixing it with --, followed by the name you choose for the variable.
   * CSS variables are usually defined inside a selector (commonly :root for global variables).
2. Using a CSS Variable:
   * To use a CSS variable, you reference it with var(--variable-name).
   * This allows you to reuse the same value multiple times within the CSS file.
3. Scope of CSS Variables:
   * If you define a variable inside a specific selector (e.g., a class or ID), the variable is only accessible within that selector.
   * Variables defined in :root are globally accessible across the entire document.
4. Changing CSS Variable Values:
   * You can modify the value of a CSS variable using JavaScript or by redefining it within specific selectors, allowing for dynamic theming or responsive design.

