CSS ASSIGNMENT

CSS Selectors & Styling

1) *What is a CSS selector? Provide examples of element, class, and ID selectors.*

Ans) A CSS selector is a pattern used to select and style HTML elements.

Examples:

I. **Element Selector** (targets all <p> elements):

```
p {
  color: blue;
}
```

II. **Class Selector** (targets elements with class. box):

```
.box {
  background-color: yellow;
}
```

III. **ID Selector** (targets element with ID #header):

```
#header {
  font-size: 20px;
}
```

IV. Universal Selector (*) Applies styles to **all elements** on the page.

```
* {
  margin: 0;
  padding: 0;
}
```

V. Group Selector (A, B, C): Styles **multiple elements** at once.

```
h1, p, button {
  font-family: Arial;
}
```

2) *Explain the concept of CSS specificity. How do conflicts between multiple styles get resolved?*

Ans) CSS Specificity & Conflict Resolution

CSS specificity determines which rule applies when multiple styles target the same element.
Specificity Calculation Order:

1. **Inline styles** (style="color: red;") → **1000**
2. **ID selectors** (#header {}) → **100**
3. **Class, attribute, and pseudo-class selectors** (.box,

[type="text"], :hover) → **10**

4. **Element selectors** (p, h1, div) → **1**
5. **Universal selector (*) and inherited styles → 0 Example of Conflicting Styles:**

```
p { color: blue; }        /* Specificity: 1 */
.text { color: green; }   /* Specificity: 10 */
#main { color: red; }     /* Specificity: 100 */
```

How Conflicts Are Resolved?

1. Higher specificity wins.
2. Equal specificity? The last rule in the CSS file wins.
3. important overrides all unless another important with higher specificity exists.

Example of important:

```
p { color: blue !important; }
#main { color: red; }
```

3) *What is the difference between internal, external, and inline CSS? Discuss the advantages and disadvantages of each approach.*

Ans) Difference Between Internal, External, and Inline CSS

CSS can be applied in three ways: Internal, External, and Inline. Each has its pros and cons.

1. Inline CSS

Defined directly within an HTML element using the style attribute.

✅ **Advantages:**

- Highest specificity (overrides other styles).
- Quick for small changes.

❌ **Disadvantages:**

- Hard to maintain for large projects.
- Increases HTML file size.
- Reduces reusability.

🔹 **Example:**

```
<p style="color: red; font-size: 20px;">This is inline CSS</p>
```

2. Internal CSS

Defined inside a <style> tag within the <head> of the HTML file.

✅ **Advantages:**

- Useful for single-page styling.
- No extra HTTP request (like external CSS).

❌ **Disadvantages:**

- Not reusable across multiple pages.
- Can increase page load time if CSS is large.

🔹 **Example:**

```
<head>
  <style>
    p {
      color: blue;
      font-size: 18px;
    }
  </style>
</head>
```

3. External CSS

Stored in a separate .css file and linked to HTML using <link>.

✅ **Advantages:**

- Best for large projects (maintainability & reusability).
- Reduces HTML file size.
- Allows caching for faster load times.

❌ **Disadvantages:**

- Requires an extra HTTP request (affects initial load speed).
- Styles won't apply if the CSS file is missing or not loaded properly.

🔹 **Example:**

html

```html
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

css

```css
p {
  color: green;
  font-size: 16px;
}
```

CSS Box Model

Explain the CSS box model and its components (content, padding, border, margin). How does each affect the size of an element?

Ans) CSS Box Model & Its Components

The CSS Box Model defines how elements are structured and spaced on a webpage. It consists of four parts:

1. Content – The actual text, image, or element inside the box.
2. Padding – Space inside the border, around the content.
3. Border – The boundary around the element.
4. Margin – Space outside the border, separating elements.

How It Affects Element Size?

Total width/height = Content + Padding + Border + Margin 🔹 Example:

```css
div {
  width: 200px;
  padding: 10px;
  border: 5px solid black;
  margin: 20px;
}
```

 Final width = 200 + 10 + 10 + 5 + 5 = 230px Final height works the same way.

5) What is the difference between border-box and contentbox box-sizing in CSS? Which is the default?

Ans) Difference Between border-box and content-box in box-sizing

1. **content-box (Default)** o Width/height only includes content. o Padding & border are added separately, increasing the total size.

```css
div {
  width: 200px;
  padding: 10px;
  border: 5px solid black;
  box-sizing: content-box; /* Default */
}
```

 Final width = 200 + 10 + 10 + 5 + 5 = 230px

2. Border-box

- Width/height **includes padding & border**.
- The content shrinks to fit within the set size.

```css
div {
  width: 200px;
  padding: 10px;
  border: 5px solid black;
  box-sizing: border-box;
}
```

- Final width = 200px (no extra size added).

CSS Flexbox

What is CSS Flexbox, and how is it useful for layout design? Explain the terms flex-container and flex-item

Ans) CSS Flexbox & Its Importance

Flexbox (Flexible Box) is a CSS layout model that helps design responsive, one-dimensional layouts efficiently by distributing space within a container.

Key Terms:

1. **Flex Container**

- The parent element that holds flex items.
- Defined using display: flex;

```css
.container {
  display: flex;
  justify-content: center;
}
```

2. **Flex Item**

- The child elements inside the flex container.
- They automatically adjust based on the container's rules.

```css
.item {
  flex: 1; /* Makes items flexible */
}
```

7) Describe the properties justify-content, align-items, and flex-direction used in Flexbox.

Ans) Key Flexbox Properties

1. justify-content (Aligns items horizontally along the main axis)
   - flex-start (default) | flex-end | center | space-between | space-around | space-evenly

```css
.container {
  justify-content: center; /* Centers items horizontally */
}
```

2. **align-items** (Aligns items **vertically** along the cross axis)
    - stretch (default) | flex-start | flex-end | center | baseline

```css
.container {
  align-items: center; /* Centers items vertically */
}
```

3. **flex-direction** (Defines the main axis direction)
    - row (default) | row-reverse | column | column-reverse

```css
.container {
  flex-direction: column; /* Stacks items vertically */
}
```

CSS Grid

Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?

Ans) CSS Grid vs. Flexbox

CSS Grid is a layout system for designing two-dimensional (rows & columns) layouts, while Flexbox is for one-dimensional (row OR column) layouts.

Key Differences:

Feature	**CSS Grid**	**Flexbox**
Layout Type	2D (rows & columns)	1D (row OR column)

Alignment	Grid-based (explicit rows/columns)	Flow-based (flexible items)
Item Control	Precise placement using grid-template-areas	Content-driven, adjusts dynamically
Best for	Complex layouts (dashboards, cards, forms)	Simple alignment (navbars, buttons)

9) Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide examples of how to use them.

Ans) Key CSS Grid Properties

1. grid-template-columns – Defines column sizes.

```css
.grid {
  display: grid;
  grid-template-columns: 100px 200px auto;
}
```

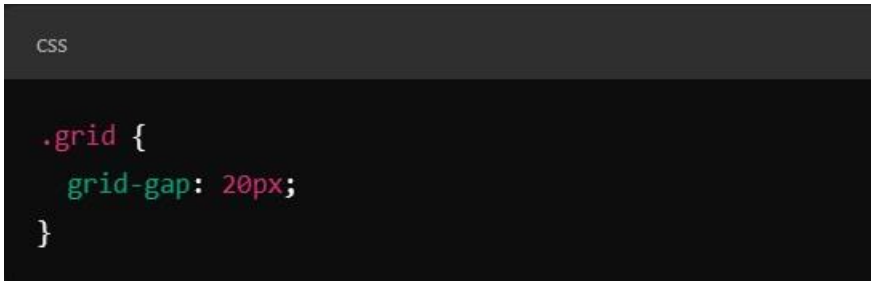
Creates 3 columns: 100px, 200px, and flexible auto-width.

2. grid-template-rows – Defines row sizes.

```css
.grid {
  grid-template-rows: 100px 150px auto;
}
```

Creates 3 rows: 100px, 150px, and flexible auto-height.

3. grid-gap (or gap) – Sets space between rows & columns.



Adds a 20px gap between all grid items.

v Responsive Web Design with Media Queries

10) *What are media queries in CSS, and why are they important for responsive design?*

Ans) Media Queries in CSS:

Media queries allow CSS to adapt styles based on screen size, resolution, or device type, making websites responsive.

Why Are They Important?

✅ Adjust layouts for different devices (mobile, tablet, desktop).

✅ Improve user experience & accessibility.

✅ Enable a **mobile-first** or **desktop-first** approach.

Example:

```css
/* Styles for screens smaller than 600px */
@media (max-width: 600px) {
  body {
    background-color: lightgray;
  }
}
```

11) *Write a basic media query that adjusts the font size of a webpage for screens smaller than 600px.*

Ans) Basic Media Query for Font Size Adjustment:

```css
@media (max-width: 600px) {
  body {
    font-size: 14px;
  }
}
```

Effect: If the screen width is **600px or smaller**, the font size will shrink to **14px**, improving readability on smaller devices.

v **Typography and Web Fonts**

12) *Explain the difference between web-safe fonts and custom web fonts. Why might you use a web-safe font over a custom font?*

Ans) Web-Safe Fonts vs. Custom Web Fonts

1. Web-Safe Fonts
   - Pre-installed on most devices (e.g., Arial, Times New Roman, Verdana).
   - Faster loading, no extra downloads.
   - Limited style choices.

2. Custom Web Fonts (e.g., Google Fonts, Adobe Fonts) o Loaded from external sources using @font-face. o More design flexibility, but slower loading if not optimized.

✅ Use Web-Safe Fonts for speed & compatibility.

✅ Use Custom Fonts for unique branding & aesthetics.

13) *What is the font-family property in CSS? How do you apply a custom Google Font to a webpage?*

Ans) CSS font-family Property

The font-family property specifies the font for text. You can list multiple fonts as fallbacks.

- Example:

```css
p {
  font-family: Arial, sans-serif;
}
```

Applying a Google Font:

Import in <head>:

- <link href="https://fonts.googleapis.com/css2?family=Roboto:w
ght@400;700&display=swap" rel="stylesheet"> Apply in CSS:
- body { font-family: 'Roboto', sans-serif; }