Lab Session 3: Initiation to CSS Stylesheets

I. General Principles

1. Understanding CSS Rules

- A **CSS rule** consists of a **selector** and a **declaration**. Example: This rule sets all <h4> elements to display in blue: **h4 { color: blue; }**
- **Structure of a CSS rule:**
  - **Selector** → Targets the HTML element (h4).
  - **Declaration** → Defines the styling (color: blue).
  - A declaration consists of:
    - **Property** (color).
    - **Value** (blue).
- **Multiple Declarations:** Use semicolons (;) to separate them:

  p { margin-left: 1cm; font-style: italic; }

- **Grouping Selectors:** If the same style applies to multiple elements, they can be grouped:

  h1, p { color: red; }, In this case, all <h1> headers and <p> paragraphs will be red.

---

2. Adding a Stylesheet to an HTML Page

There are **three** ways to apply CSS:

1. Inline CSS (Directly in an Element) – Quick but Not Recommended

```
<p style="color: blue; font-size: 14px;">This is a test.</p>
```

- **Pros:** Fast and easy for small tweaks.
- **Cons:** Difficult to manage for large projects.

2. Internal CSS (Within the <head> of the Document)

```
<head>
  <style>
    h1, p { color: red; }
    p { margin-left: 1cm; font-style: italic; }
  </style>
</head>
```

- **Pros:** Keeps styles separate from content.
- **Cons:** Still embedded in the document, making it harder to reuse styles across multiple pages.

3. External CSS (Recommended for Large Projects)

- Create a file named styles.css and link it in the <head>:
- <link rel="stylesheet" href="styles.css">

Pros: Best for maintainability, One CSS file can style multiple pages.

body
{ background-color: #f0f0f0;
font-family: Arial, sans-serif;
margin: 20px; padding: 20px; }

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

p { color: darkgray; font-size: 16px; }

II. CSS Classes

1. Classes allow you to define reusable styles

- **Defining a Class:** Use a dot (.) before the class name in CSS.

.highlight {color: yellow; font-weight: bold ;}

- **Applying a Class to an HTML Element:** This text is highlighted.

2. Applying Multiple Classes to an Element

Separate class names with a space:

This is styled with multiple classes.

- **Corresponding CSS:**

.highlight {color: yellow;}

.large-text {font-size: 20px;}

III. CSS Specificity & Priority

1. The Importance of CSS Order

- If multiple styles are applied to the same element, the **most specific** or **latest** definition wins.

Example of Overriding Styles

```
p { color: brown;}
.special { color: green;  font-weight: bold;
}
<p class="special">This text will be green and bold.</p>
```

- Since the .special class is **more specific** than the p selector, it **overrides** the color: brown; rule.

2. Handling Conflicts in CSS

When multiple conflicting rules exist, CSS follows these principles:

1. **Inline styles** (style="...") override everything.
2. **ID selectors (#id)** are stronger than **classes (.class)**.
3. **Classes (.class)** are stronger than **element selectors (p, h1, div, etc.)**.
4. **Later rules in the CSS file** take precedence over earlier ones.

Example: Specificity Levels

```
p { color: red; }           /* Low specificity */
.special { color: blue; }    /* Higher specificity */
#important { color: green; } /* Highest specificity */
<p id="important" class="special">This text will be green.</p>
```

- The **ID selector** #important has the highest priority, so the text is **green**.

IV. Nested Elements & Inheritance

1. How Styles Are Passed to Child Elements

Some styles **inherit** (e.g., color, font-family), while others **do not** (e.g., margin, border). Example

```
body {
  color: blue;
}
<body>
  <p>This text will be blue.</p> <!-- Inherited from <body> -->
</body>
```

2. Nesting Example

```
div { color: red;}

p {  color: green;}
<div>
  <p>This text will be green, not red.</p>
</div>
```

Even though is red, <p> explicitly sets **green**, so it **overrides** the inherited color.

V. The "Cascade" Concept in CSS

1. What Is the CSS Cascade?

- The cascade determines **which styles take effect** when multiple rules apply.
- The order of styles matters—**later styles override earlier ones**.

2. Using Multiple Stylesheets

You can **combine multiple stylesheets**, but **the last one loaded takes precedence**.

Example: Combining External Stylesheets

```
<head>
  <link rel="stylesheet" href="style1.css">
  <link rel="stylesheet" href="style2.css">
  <style>
            h1 { color: fuchsia; }
            h2 { font-size: 16pt; }
            h3 { font-size: 14pt; }
  </style>
</head>
```

- If **style2.css** has rules conflicting with **style1.css**, the **second file's rules take precedence**.
- The **<style> block in the <head>** takes precedence over both external stylesheets.

VI. Best Practices for Writing Clean & Maintainable CSS

✔ **Use External Stylesheets:** Keeps your HTML clean and easy to maintain.
✔ **Use Classes & IDs Properly:** Classes for reusable styles, IDs for unique elements.
✔ **Avoid Inline Styles:** They make code harder to read and maintain.
✔ **Keep Specificity Low:** Use classes over IDs to keep styles flexible.
✔ **Organize Your Stylesheets:** Group similar properties together and use comments.
✔ **Test on Different Browsers:** CSS may render differently in Chrome, Firefox, Edge, etc.
