**Lab 0:Foundation & Styling**

**Objective:** Understand different ways to apply CSS.

* **Exercise:**
  + Create an HTML page with h1, p, a (link), and img tags.
  + Apply inline CSS to change the color of one paragraph.
  + Apply internal CSS to style all h1 elements with a specific font and color.
  + Create an external CSS file and link it to the HTML page. Style all p elements with a different font family and line-height.

**Lab 1: CSS Selectors & Basic Styling**  
**Objective:** Learn to target elements via element, class, id, attribute, pseudo‑class, and pseudo‑element selectors.

* **Exercise:**
  1. Create an HTML page with headings, paragraphs, lists, and a <div> with a unique id.
  2. Use element selectors to color all <p> tags.
  3. Use class selectors to style a set of <li> items differently.
  4. Use an attribute selector to style all <a> tags whose href contains “http.”
  5. Demonstrate a pseudo‑class (:hover) and pseudo‑element (::first-letter) rule.
  6. **Use the :nth-child(even) pseudo-class** to style every even <li> in a list with a different background color.
  7. **Use the ::before pseudo-element** on a heading (e.g., <h2>) to insert a decorative symbol or label before the text.

**Lab 2: The CSS Box Model & Layout Fundamentals**  
**Objective:** Visualize and manipulate content, padding, border, and margin; practice block vs inline.

* **Exercise:**
  1. Create three adjacent <div>s with colored borders.
  2. Adjust padding/margin to control spacing.
  3. Use display: inline-block vs block to observe behavior.
  4. Experiment with border-radius to create rounded corners.
  5. Use box-sizing: border-box and box-sizing: content-box to observe their effects on element sizing.

**Lab 3: Advanced Positioning & Float**  
**Objective:** Use position (static, relative, absolute, fixed) and float to create simple layouts.

* **Exercise:**
  1. Float an image left of a paragraph of text.
  2. Create a two‑column layout: fixed sidebar (absolute) + fluid content area.
  3. Implement a sticky header using position: fixed.

**Lab 4: Navigation Bar & Responsive Media Queries**  
**Objective:** Build a horizontal navbar, then make it responsive via media queries.

* **Exercise:**
  1. Create a <ul>‑based navbar styled with display: inline-block.
  2. At ≤ 600 px viewport width, switch to a vertical (stacked) menu using @media.

**Lab 5: CSS Transitions & Animations**  
**Objective:** Add hover‑transitions and a keyframe animation.

* **Exercise:**
  1. On hover, smoothly change a button’s background over 0.3 s.
  2. Define a @keyframes rule to animate an element’s transform: rotate() indefinitely.

**Lab 6: Layouts with Floats and Flexbox (or Grid)**

* **Objective:** Implement different page layouts using modern CSS techniques.
* **Exercise:**
  + **Option A (Floats):** Design a two or three-column layout for a simple blog page using float properties. Include a header, navigation, content area, sidebar, and footer. Clear floats appropriately.
  + **Option B (Flexbox):** Design a responsive navigation bar using Flexbox, where menu items are evenly spaced and stack vertically on smaller screens.
  + **Option C (Flexbox & Grid):** Create a simple photo gallery using CSS Grid for the overall layout and Flexbox for individual image captions/details.

**Lab 7: Text and Font Properties**

* **Objective:** Control text appearance and typography.
* **Exercise:**
  + Create an HTML page with various headings and paragraphs.
  + Experiment with font-family, font-size, font-weight, font-style.
  + Apply text-align, text-decoration, text-transform.
  + Demonstrate letter-spacing and word-spacing.
  + Use Google Fonts to integrate a custom font into your page.