CSS RULES AND CONFLICT RESOLUTION

STYLE PLACEMENT:

- Our placement of styles affects which style declarations override which ones.
- We can place our styling in three places:
    - In the <head> tag, we can place styling using <style> tag.
    - Directly with the element using **“style”** attribute.
    - Place all the styles in an external stylesheet.

- We specify external styles using:
    - **<link> tag. We specify two attributes:**
    - **rel = ‘stylesheet’, which tells the browser that it is a stylesheet.**
    - **href = we provide link of the external CSS.**

- Using external styles, we can have reusability and consistency in all HTML pages.
- Head styles are used to override external CSS.

CONFLICT RESOLUTION:

- There can be a conflict when we define styling in multiple ways, we have to decide which rules would be given priority.

Cascading Algorithm:

- It is an algorithm defining how to combine properties values originating from multiple different sources.
- Cascade combines the importance, origin, specificity and source order of the applicable style declarations to determine which declarations should be applied.

1. ORIGIN PRECEDENCE:
    - When two declarations are in conflict i.e., they specify the same property for the same target. Origin precedence comes with a rule that **“LAST DECLARATION WINS”.**
    - When different declarations don’t conflict i.e., they specify different property for the same target. The rules is **“DECLARATIONS ARE MERGED”.**

2. INHERITANCE:
    - Basic idea is that we have document object model tree. There is <body> having some element within it, and those elements can also have element within in.
    - If we specify styles for a parent tag, all children of that parent will inherit those styles.
    - It does not work in reverse i.e., if we apply to a child, the same styles won’t be applied to the parent. Can apply common styles in **<body>** as they are applied to the whole page.

3. SPECIFICITY:
   - The rule is **MOST SPECIFIC SELECTOR COMBINATION WINS.**
   - We can calculate the specificity of a selector by a **score**, the highest would indicate that it is the most specific combination.
   - Arrange types of things that affect score from left to right (highest to lowest):
     - **style = "…"   ID   class, pseudo-class, attribute   #no of elements**
     - Style is the most specific as it is specified with the tag/element.
     - The least is the number of elements used in the selector combination.

- We can override all the rules using **!important** when defining styles.
- For e.g., **p { color: green !important; }**
- Avoid using this though.

STYLING TEXT:

- **font-family:**
  - We specify more than 1 font families like: 'Times New Roman', Times, sarif.
  - In the very end, we select the default font to be selected (either sarif or san-sarif).
  - We specify more than once because it might be that some font in our desktop may not work for some other desktop if it isn't on their computer.
- **color:**
  - Use hex value.
- **font-style:**
  - If we want italic or normal, etc.
- **font-weight:**
  - Weight of boldness of font, can specify keyword **"bold"** or a numeric value.
- **font-size:**
  - We use pixels (px) which is absolute unit of measurement of size.
  - Pixels are relative to the viewing device.
- **text-transform:**
  - Control how text looks: uppercase, lowercase, etc.
- **text-align:**
  - Allows to be center, left, right, justify (within its block element).

- **Relative Font Sizing:**
  - There are 2 units for this: **percent, ems.**
  - **Percent:**
    - font-size = 120% means that we want to take the default size and increase it by 120%
    - Usually used in **<body>** to increase same size for all the HTML elements.
  - **EM:**
    - Unit of measurement equivalent to width of the letter 'm' in the font we use.
    - It is relative to whatever the current font-size is.
    - font-size = 2em, takes the default size and gives the twice size of it.