Topic/Title:

CSS Specificity || Combining CSS selectors || CSS Positioning ||

Keywords/Questions:

Notes:

CSS specificity determines which styles are applied to an element when there are conflicting styles. It's based on:

1. Inline styles: Highest specificity, applied directly to an element.

2. ID selectors: Higher specificity than classes or tag selectors (#elementID).

3. Class selectors, attribute selectors, pseudo-classes: Same specificity (.class, [type="text"], :hover).

4. Tag selectors: Lowest specificity (div, a, p).

5. Universal selector (*): Lowest specificity, applies to all elements.

These are key concepts in CSS that influence how styles are applied:

1. Position: Determines how elements are positioned within the document flow.
   Values: static, relative, absolute, fixed, sticky.

2. Specificity: Determines which CSS rule takes precedence when multiple rules apply to the same element.
   Calculated based on the types of selectors used (inline styles > ID selectors > class/attribute selectors > type selectors).

3. Type: Refers to the different types of CSS rules and their application:
   Inline styles: Applied directly to an element with the style attribute.
   Internal styles: Defined within the <style> element in the <head> section of an HTML file.
   External styles: Linked using the <link> element to an external CSS file.

4. Importance: Specifies the priority of a CSS rule, overriding normal specificity rules.
   !important keyword: Applied to a CSS declaration to give it the highest priority.

There are five different position values: 1. static 2. relative 3. fixed 4. absolute 5. sticky

Summary:

To know more details about specificity go to this: https://www.w3schools.com/css/css_specificity.asp

Static - this is the default value, all elements are in order as they appear in the document.
Relative - the element is positioned relative to its normal position.
Absolute - the element is positioned absolutely to its first positioned parent.
Fixed - the element is positioned related to the browser window.

Sticky: A combination of relative and fixed positioning, where the element acts like a relative element until it reaches a certain scroll point, at which point it acts like a fixed element.

1. Descendant Selector

Targets elements that are descendants of a specified element.

```
/* Targets all <p> elements that are inside <div> elements */
div p {
  color: blue;
}
```

2. Child Selector

Targets elements that are direct children of a specified element.

```
/* Targets all direct <li> children of <ul> */
ul > li {
  list-style-type: none;
}
```

3. Adjacent Sibling Selector

Targets an element that is immediately preceded by a specified element.

```
/* Targets the first <p> that follows an <h2> */
h2 + p {
  margin-top: 0;
}
```

4. General Sibling Selector

Targets all sibling elements that follow a specified element.

```
/* Targets all <p> elements that follow any <h2> */
h2 ~ p {
  color: green;
}
```

5. Group Selector

Applies the same style to multiple elements.

```
/* Targets <h1>, <h2>, and <h3> with the same style */
h1, h2, h3 {
  font-family: Arial, sans-serif;
}
```

6. Attribute Selector

Targets elements based on their attributes or attribute values.

```
/* Targets all <input> elements with a type attribute of "text" */
input[type="text"] {
  border: 1px solid #ccc;
}

/* Targets all <a> elements with a title attribute */
a[title] {
  text-decoration: underline;
}
```

7. Pseudo-Class Selector

Targets elements based on their state or position.

```
/* Targets the first <li> in each <ul> */
ul li:first-of-type {
  font-weight: bold;
}

/* Targets <a> elements when hovered over */
a:hover {
  color: red;
}
```

8. Pseudo-Element Selector

Targets specific parts of an element.

```
/* Adds content before <p> elements */
p::before {
  content: "Note: ";
  font-weight: bold;
}
```

9. Combining Multiple Selectors

You can combine different types of selectors for more precise targeting.

```
/* Targets <a> elements that are inside <nav> and have a class of "active" */
nav a.active {
  color: red;
}

/* Targets <p> elements that are both direct children of <section> and have a class of "highlight" */
section > p.highlight {
  background-color: yellow;
}
```