**HW4**

1) (i) **max-width**: The max-width property is used to set the maximum width of an element. This prevents the value of the width property from becoming larger than max-width.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

p {

max-width: 300px;

background-color: pink;

}

</style>

</head>

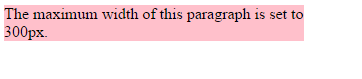
<body>

<p>The maximum width of this paragraph is set to 300px.</p>

</body>

</html>

**Output:**

****

(ii) **min-width**: The min-width property is used to set the minimum width of an element. This prevents the value of the width property from becoming smaller than min-width.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

p {

min-width: 150px;

background-color: orange;

}

</style>

</head>

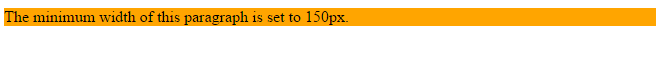
<body>

<p>The minimum width of this paragraph is set to 150px.</p>

</body>

</html>

**Output:**



2) (i) **pseudo-element**: A CSS pseudo-element is used to style specified parts of an element.

It can be used to:

* Style the first letter, or line, of an element
* Insert content before, or after, the content of an element

**Syntax:**

selector::pseudo-element {  
    property:value;  
}

(ii) **pseudo-class**: A pseudo-class is used to define a special state of an element.

It can be used to:

* Style an element when a user mouses over it
* Style visited and unvisited links differently
* Style an element when it gets focus

**Syntax:**

selector:pseudo-class {  
    property:value;  
}

3) (i) **box-sizing**: The box-sizing property is used for the sizing properties like width and height.

**Example:**

<!DOCTYPE html>

<html>

<head>

<style>

.div1 {

width: 300px;

height: 100px;

padding: 50px;

border: 1px solid red;

}

.div2 {

width: 300px;

height: 100px;

padding: 50px;

border: 1px solid blue;

box-sizing: border-box;

}

</style>

</head>

<body>

<h2>Without box-sizing</h2>

<div class="div1">This div is bigger (width is also 300px and height is 100px).</div>

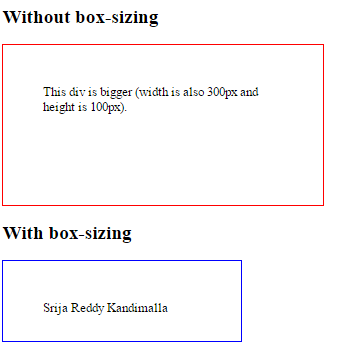
<h2>With box-sizing</h2>

<div class="div2">Srija Reddy Kandimalla</div>

</body>

</html>

**Output:**



(ii) **Display property values**:

Inline: Default value. Displays an element as an inline element (like <span>)

Block: Displays an element as a block element (like <p>)

Flex: Displays an element as an block-level flex container. New in CSS3

inline-block: Displays an element as an inline-level block container. The inside of this block is formatted as block-level box, and the element itself is formatted as an inline-level box

inline-flex: Displays an element as an inline-level flex container.

inline-table: The element is displayed as an inline-level table

list-item: Let the element behave like a <li> element

run-in: Displays an element as either block or inline, depending on context

table: Let the element behave like a <table> element

none: The element will not be displayed at all (has no effect on layout)

initial: Sets this property to its default value.

Inherit: Inherits this property from its parent element.

4) Differences between display none and visibility hidden property

|  |  |
| --- | --- |
| Display none | visibility hidden |
| * **display**:**none** will not be available in the page * **display**:**none** doesn't preserve the space   **Example:**  <!DOCTYPE html>  <html>  <head>  <style>  h1.hidden {  display: none;  }  </style>  </head>  <body>  <h1>This is a visible heading</h1>  <h1 class="hidden">This is a hidden heading</h1>  <p>Notice that the h1 element with display: none; does not take up any space.</p>  </body>  </html>  **Output:** | * **visibility**:**hidden** hides an element, but it will still take up the same space as before. The element will be **hidden**, but still affect the layout * **visibility**:**hidden** preserve the space   **Example:**  <!DOCTYPE html>  <html>  <head>  <style>  h1.hidden {  visibility: hidden;  }  </style>  </head>  <body>  <h1>This is a visible heading</h1>  <h1 class="hidden">This is a hidden heading</h1>  <p>Notice that the hidden heading still takes up space.</p>  </body>  </html>  **Output:** |