



NOTES

Advance CSS -1



Advance CSS

1. More CSS Selectors

1.1 Combinator selectors (space, >, +, ~)

CSS combinators define relationships between selectors. They include descendant (), child (>), adjacent sibling (+), and general sibling (~), allowing targeted selection of elements based on their hierarchy.

>C~S+S Combinators

1.1.1 Descendant combinator (space)

The descendant combinator in CSS is used to select elements that are nested within other elements. It is represented by a space between two selectors. This combinator matches all descendants (children, grandchildren, etc.) of a specified element.

Sample Snippet:

```
div h1 {  
  font-size: 40px;  
  color: purple;  
}
```

Here's an example of combinator selectors in CSS :-

HTML:

```
Unset  
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <title></title>
```

```
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">

</head>
<body>
  <div>
    <p>This paragraph 1 is inside a div.</p>
    <span>This span is inside a div.</span>
  </div>

  <p>This paragraph 2 is outside the div.</p>

</body>
</html>
```

CSS:

```
Unset
div span {
  color: blue;
  background-color: burlywood;
}
```

Explanation:

- The CSS rule `div p { color: blue; }` applies to all `` elements inside any `<div>`, no matter how deeply nested.
- In this case, all spans inside the `<div>` will be styled with blue text, but the span outside the `<div>` remains unaffected.

1.1.2 Child combinator (>)

The child combinator in CSS selects elements that are direct children of a specified element. It is represented by the `>` symbol placed between two selectors. This combinator matches only the immediate child elements, not any nested descendants.

Sample Snippet:

Unset

```
div > p {  
    background-color: yellow;  
}
```

Let's discuss the above situation with example:

HTML :

Unset

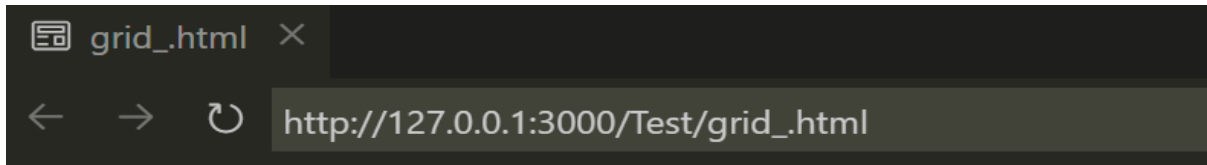
```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <title></title>  
    <meta name="description" content="">  
    <meta name="viewport" content="width=device-width, initial-scale=1">  
    <link rel="stylesheet" href="style11.css">  
  
  </head>  
  <body>  
    <div>  
      <p>This paragraph 1 is inside a div.</p>  
      <span>This span is inside a div.</span>  
      <p>This paragraph 2 is inside a div.</p>  
      <p>This paragraph 3 is inside a div.</p>  
      <p>This paragraph 4 is inside a div.</p>  
    </div>  
  
    <p>This paragraph 2 is outside the div.</p>  
  
  </body>  
</html>
```

CSS:

Unset

```
div > p {  
    color: blue;  
    background-color: burlywood;  
}
```

OUTPUT:



This paragraph 1 is inside a div.

This span is inside a div.

This paragraph 2 is inside a div.

This paragraph 3 is inside a div.

This paragraph 4 is inside a div.

This paragraph 2 is outside the div.

1.1.3 Next sibling combinator (+)

The next sibling combinator in CSS selects an element that immediately follows a specified element. It is represented by the + symbol and targets only the first sibling that comes after the specified element.

Sample Snippet:

```
Unset
div + p {
  background-color: yellow;
}
```

Let's learn the use of next sibling combinator with example:

HTML :

```
Unset
<!DOCTYPE html>
<html>
  <head>
```

```
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<title></title>
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">

</head>
<body>
  <div>
    <p>This paragraph 1 is inside a div.</p>
    <span>This span is inside a div.</span>
    <p>This paragraph 2 is inside a div.</p>
    <p>This paragraph 3 is inside a div.</p>
    <p>This paragraph 4 is inside a div.</p>
  </div>

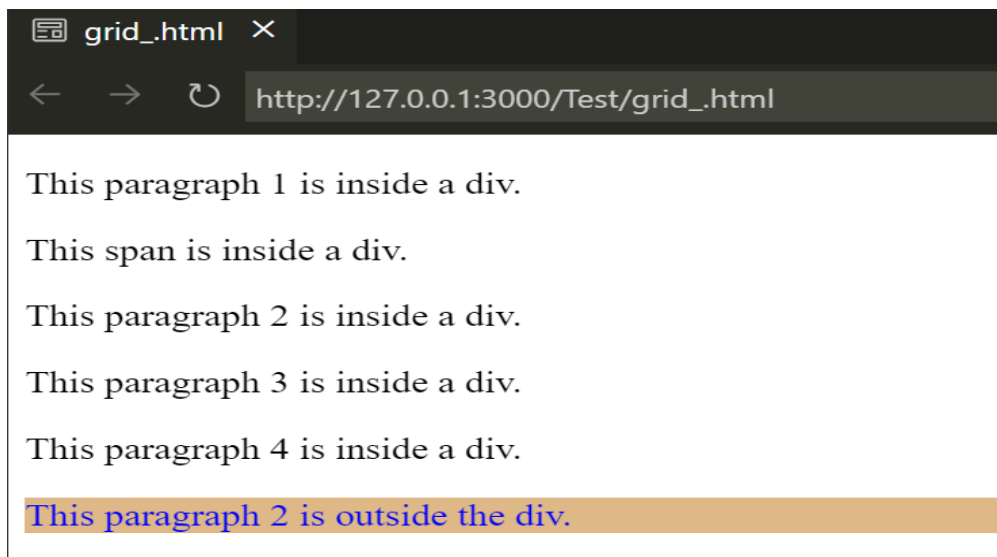
  <p>This paragraph 2 is outside the div.</p>

</body>
</html>
```

CSS:

```
Unset
div + p {
  color: blue;
  background-color: burlywood;
}
```

OUTPUT:



1.1.4 Subsequent-sibling combinator (~)

The subsequent sibling combinator in CSS selects all siblings of a specified element that come after it. Represented by the `~` symbol, it allows you to style all following siblings that share the same parent.

Let us take an example to demonstrate the use of Subsequent Sibling Combinator.

Sample Snippet:

```
Unset
div ~ p {
  background-color: yellow;
}
```

Example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Combinators</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<link rel="stylesheet" href="style11.css">

</head>
<body>
  <div>
    <p>This paragraph 1 is inside a div.</p>
    <span>This span is inside a div.</span>
    <p>This paragraph 2 is inside a div.</p>
    <p>This paragraph 3 is inside a div.</p>
    <p>This paragraph 4 is inside a div.</p>
  </div>

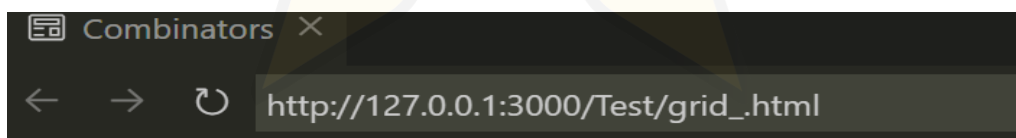
  <p>This paragraph 2 is outside the div.</p>

</body>
</html>
```

CSS:

```
Unset
span ~ p {
  color: blue;
  background-color: burlywood;
}
```

OUTPUT:



This paragraph 1 is inside a div.

This span is inside a div.

This paragraph 2 is inside a div.

This paragraph 3 is inside a div.

This paragraph 4 is inside a div.

This paragraph 2 is outside the div.

1.2 Attribute selector

The CSS attribute selector lets you style HTML elements based on their attributes and values. It uses square brackets ([]) to target elements with specific attributes.

There are various ways to use attribute selectors in CSS:

1.2.1. [attribute]

This attribute selector selects elements that have the specified attribute, regardless of its value.

Syntax:

```
Unset
/* Selects all elements with a "target" attribute */

[target] {
    background-color: yellow;
}
```

Let us consider an example for demonstration of attribute selector property:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">

  </head>
  <body>
    <div>
      <p>This <a href="#">paragraph</a> 1 is inside a div.</p>
      <span>This span is inside a div.</span>
      <p>This paragraph 2 is inside a div.</p>
      <p>This paragraph 3 is inside a div.</p>
      <p>This paragraph 4 is inside a div.</p>
    </div>
```

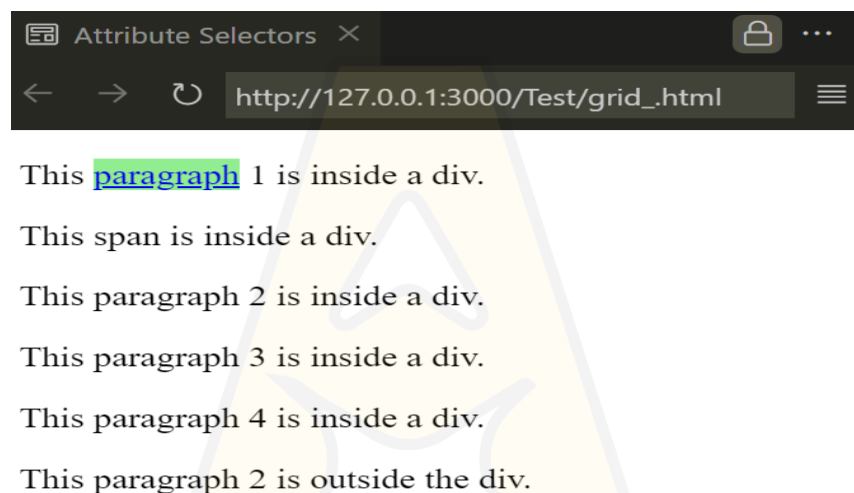
```
<p>This paragraph 2 is outside the div.</p>

</body>
</html>
```

CSS:

```
Unset
a[href] {
  background-color: lightgreen;
}
```

OUTPUT:



1.2.2. [attribute="value"]

This selector selects all the attribute elements with an attribute that exactly matches a specific value.

Syntax:

```
Unset
/* Selects all elements where target="blank" */

[target="blank"] {
  background-color: lightblue;
}
```

Let us consider an example for demonstration of attribute selector property by specifying value for them:

HTML:

Unset

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://google.com">paragraph</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p>This paragraph 2 is inside a div.</p>
      <p>This paragraph 3 is inside a div.</p>
      <p>This paragraph 4 is inside a div.</p>
    </div>

    <p>This paragraph 2 is outside the div.</p>

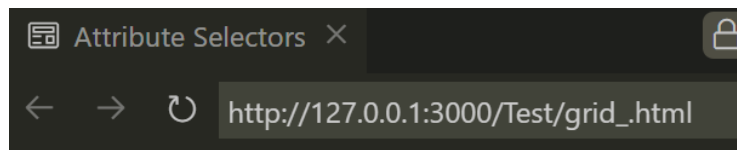
  </body>
</html>
```

CSS:

Unset

```
a[href="https://google.com"] {
  background-color: lightgreen;
}
```

OUTPUT:



This paragraph 1 is inside a div.

This span is inside a div.

This paragraph 2 is inside a div.

This paragraph 3 is inside a div.

This paragraph 4 is inside a div.

This paragraph 2 is outside the div.

1.2.3. [attribute~="value"]

Attribute selector with “ ~ ” Selects elements whose attribute value is a space-separated list, and one of the values is exactly the specified value.

Syntax:

```
Unset
/* Selects all elements where the class attribute contains the word "btn" */

[class~="btn"] {
    background-color: green;
}
```

Let us consider an example for demonstration of attribute selector property by specifying value for them:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<link rel="stylesheet" href="style11.css">
</head>
<body>
  <div>
    <p>This <a href="https://google.com">paragraph</a> 1 is inside a
div.</p>
    <span>This span is inside a div.</span>
    <p name="myName">This paragraph 3 is inside a div.</p>
    <p name="personal myName">This paragraph 4 is inside a div.</p>
  </div>

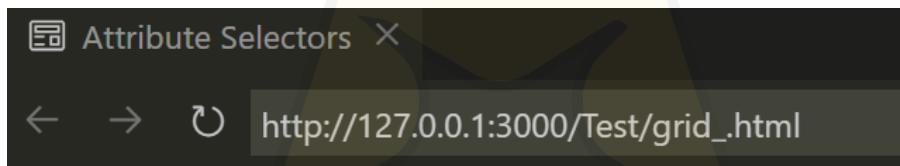
  <p>This paragraph 2 is outside the div.</p>

</body>
</html>
```

CSS:

```
Unset
p[name~="myName"] {
  background-color: lightgreen;
}
```

OUTPUT:



This [paragraph](https://google.com) 1 is inside a div.

This span is inside a div.

This paragraph 3 is inside a div.

This paragraph 4 is inside a div.

This paragraph 2 is outside the div.

1.2.4. [attribute^="value"]

Start with, “ ^ ” attribute is used to select elements whose attribute value starts with the specified value.

Syntax:

```
Unset
/* Selects all elements with href attributes that start with "https" */

[href^="https"] {
    color: green;
}
```

Let us consider an example for demonstration of attribute selector with “^” property by specifying value for them:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://google.com">paragraph</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p name="myName">This paragraph 3 is inside a div.</p>
      <p name="personal myName">This paragraph 4 is inside a div.</p>
    </div>

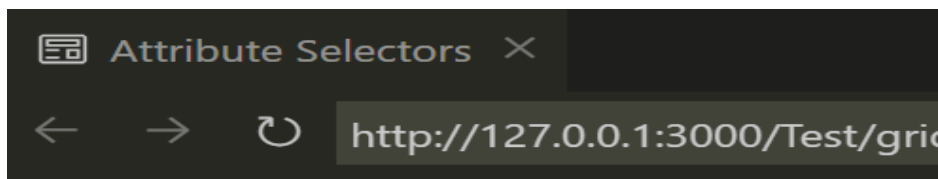
    <p>This paragraph 2 is outside the div.</p>

  </body>
</html>
```

CSS:

```
Unset
[href^="https"]{
    background-color: lightgreen;
}
```

OUTPUT:



This paragraph 1 is inside a div.

This span is inside a div.

This paragraph 3 is inside a div.

This paragraph 4 is inside a div.

This paragraph 2 is outside the div.

1.2.5. [attribute\$="value"]

Ends with attribute selector “ \$ ” is a selector whose elements whose attribute value ends with the specified value.

Syntax:

```
Unset
/* Selects all elements with href attributes that end with ".pdf" */

[href$=".pdf"] {
    color: red;
}
```

Let us consider an example for demonstration of attribute selector property by specifying value for them:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
```

```

<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">
</head>
<body>
  <div>
    <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
    <span>This span is inside a div.</span>
    <p name="myName">This paragraph 3 is inside a div.</p>
    <p>This <a href="https://amazon.com">Amazon</a> 1 is inside a div.</p>
    <p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

    <p name="personal myName">This paragraph 4 is inside a div.</p>
    <p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>
  </div>
  <p>This paragraph 2 is outside the div.</p>

</body>
</html>

```

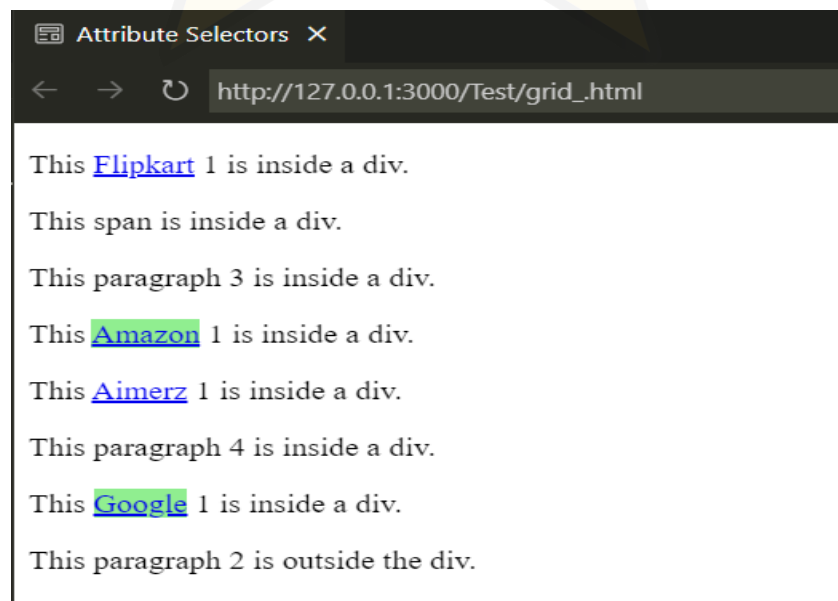
CSS:

```

Unset
[href $=".Com"]{
  background-color: lightgreen;
}

```

OUTPUT:



1.2.6. [attribute*="value"]

The “ * ” selects all the elements whose attribute value contains the specified value anywhere in our scope of declared tags.

Syntax:

```
Unset
/* Selects all elements with href attributes that contain "example" */

[href*="example"] {
    background-color: yellow;
}
```

Herein let's implement “ * ” with the attribute selector with an common example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Attribute Selectors</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p name="myName">This paragraph 3 is inside a div.</p>
      <p>This <a href="https://amazon.com">Amazon</a> 1 is inside a div.</p>
      <p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

      <p name="personal myName">This paragraph 4 is inside a div.</p>
      <p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>

    </div>

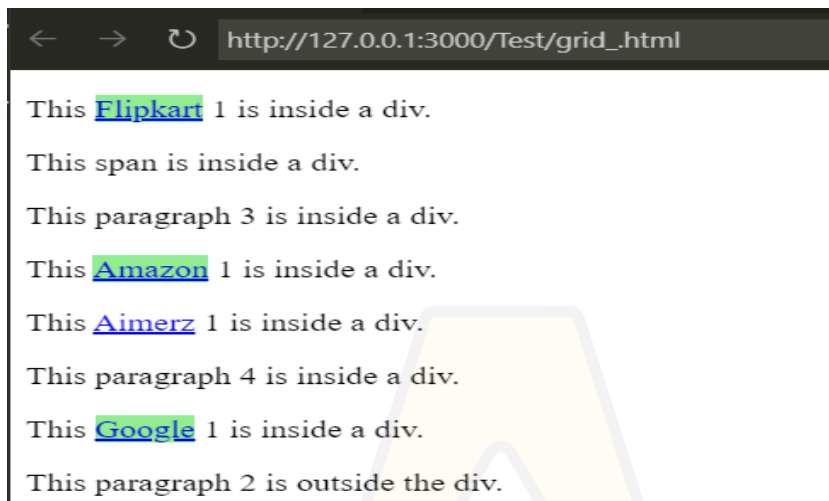
    <p>This paragraph 2 is outside the div.</p>

  </body>
</html>
```

CSS:

```
Unset
[href *= ".com"]{
    background-color: lightgreen;
}
```

OUTPUT:



In the above example you can clearly see that Aimerz website hyperlink does not end with “.com ” so the attribute selectors have not selected it for the color property change.

2. Pseudo Elements

Pseudo-elements in CSS are used to style specific parts of an our elements without the requirement to add extra HTML. They allow us to select and style parts of an element in due occasion for an instance, such as the first letter, first line, or the content before or after an element.

2.1 ::after

The ::after pseudo-element in CSS is used to insert content after the content of a selected element. It allows you to add visual cues or decorations without modifying the HTML structure.

Let us take an example to demonstrate the use of “::after“, a pseudo element in CSS.

- Let's consider a situation where we need to add a sentence after the paragraph `<p>` tag. Herein further code snippet along with the output is shown below.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Elements</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p name="myName">This paragraph 3 is inside a div.</p>
      <p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.</p>
      <p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

      <p name="personal myName">This paragraph 4 is inside a div.</p>
      <p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>

    </div>

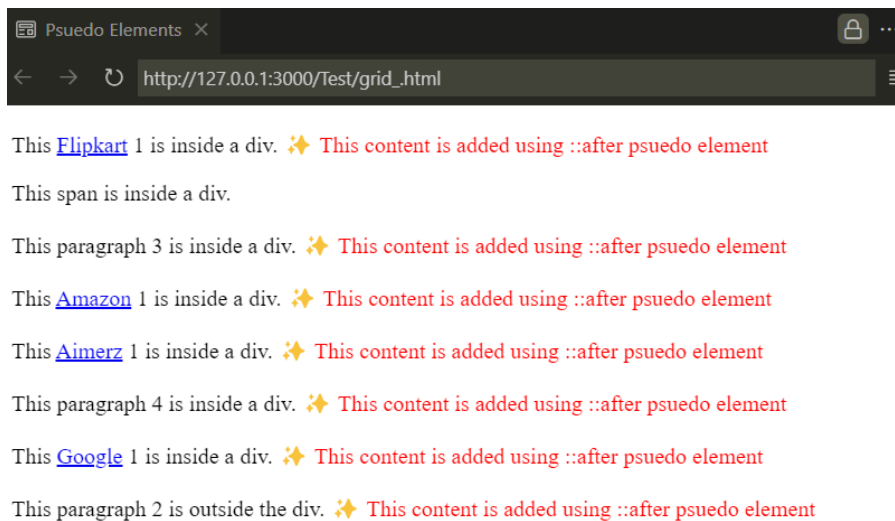
    <p>This paragraph 2 is outside the div.</p>

  </body>
</html>
```

CSS:

```
Unset
p::after {
  content: " ✨ This content is added using ::after psuedo element";
  color: red;
}
```

OUTPUT:



2.2. ::before

The ::before pseudo-element in CSS is used to insert content before the content of a selected element. It allows you to add visual cues or decorations without modifying the HTML structure.

Let us take an example to demonstrate the use of “::before“, a pseudo element in CSS.

- Let's consider a situation where we need to add a sentence before the paragraph <p> tag. Herein further code snippet along with the output is shown below.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Elements</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
```

```
<span>This span is inside a div.</span>
<p name="myName">This paragraph 3 is inside a div.</p>
<p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.</p>
<p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

<p name="personal myName">This paragraph 4 is inside a div.</p>
<p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>

</div>

<p>This paragraph 2 is outside the div.</p>

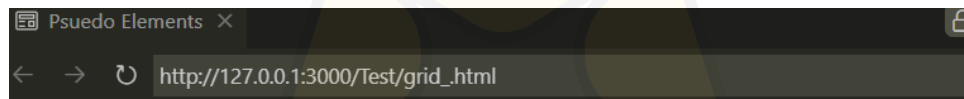
</body>
</html>
```

CSS:

Unset

```
p::before {
  content: " ✨ This content is added using ::after psuedo element";
  color: red;
}
```

OUTPUT:



✨ This content is added using ::after psuedo elementThis [Flipkart](https://flipkart.com) 1 is inside a div.
 This span is inside a div.
 ✨ This content is added using ::after psuedo elementThis paragraph 3 is inside a div.
 ✨ This content is added using ::after psuedo elementThis [Amazon](https://amazon.com) 1 is inside a div.
 ✨ This content is added using ::after psuedo elementThis [Aimerz](https://aimerz.ai) 1 is inside a div.
 ✨ This content is added using ::after psuedo elementThis paragraph 4 is inside a div.
 ✨ This content is added using ::after psuedo elementThis [Google](https://google.com) 1 is inside a div.
 ✨ This content is added using ::after psuedo elementThis paragraph 2 is outside the div.

* Above Implication is used to add the content before every occurrence of <p> paragraph tag.

2.3 ::first-letter

The ::first-letter pseudo-element in CSS is used to target and style the first letter of a block-level element, such as a paragraph (<p>) or a heading (<h1>). This is useful for typographic effects like drop caps, where the first letter is styled differently from the rest of the text.

Let's take a look at an example to see the application areas of ::first-letter property.

Example: Make the first letter of the very first word of the sentence inside <p> paragraph, tag

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Elements</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p name="myName">This paragraph 3 is inside a div.</p>
      <p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.</p>
      <p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

      <p name="personal myName">This paragraph 4 is inside a div.</p>
      <p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>

    </div>

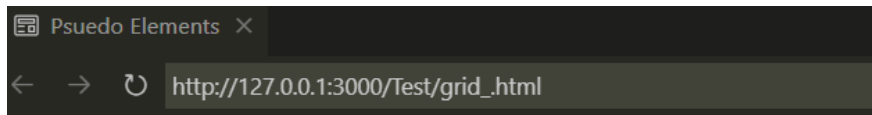
    <p>This paragraph 2 is outside the div.</p>

  </body>
</html>
```

CSS:

```
Unset
p::first-letter {
  color: red;
}
```

OUTPUT:



This [Flipkart](#) 1 is inside a div.

This span is inside a div.

This paragraph 3 is inside a div.

This [Amazon](#) 1 is inside a div.

This [Aimerz](#) 1 is inside a div.

This paragraph 4 is inside a div.

This [Google](#) 1 is inside a div.

This paragraph 2 is outside the div.

2.4 ::first-line

The ::first-line pseudo-element in CSS applies styles to the first line of a block-level element, like a paragraph or heading. The first line depends on the width of the block and how much text fits on that line, so it can change based on screen size or container width.

Example demonstration:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
<title>Psuedo Elements</title>
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">
</head>
<body>
  <div>
    <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
    <span>This span is inside a div.</span>
    <p name="myName">This paragraph 3 is inside a div.</p>
    <p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.\
    <br> This is my second line.
    </p>
    <p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

    <p name="personal myName">This paragraph 4 is inside a div.
    <br> This is my second line.
    </p>
    <p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>
    <br> This is my second line.

  </div>

  <p>This paragraph 2 is outside the div.
  <br> This is my second line.
  </p>

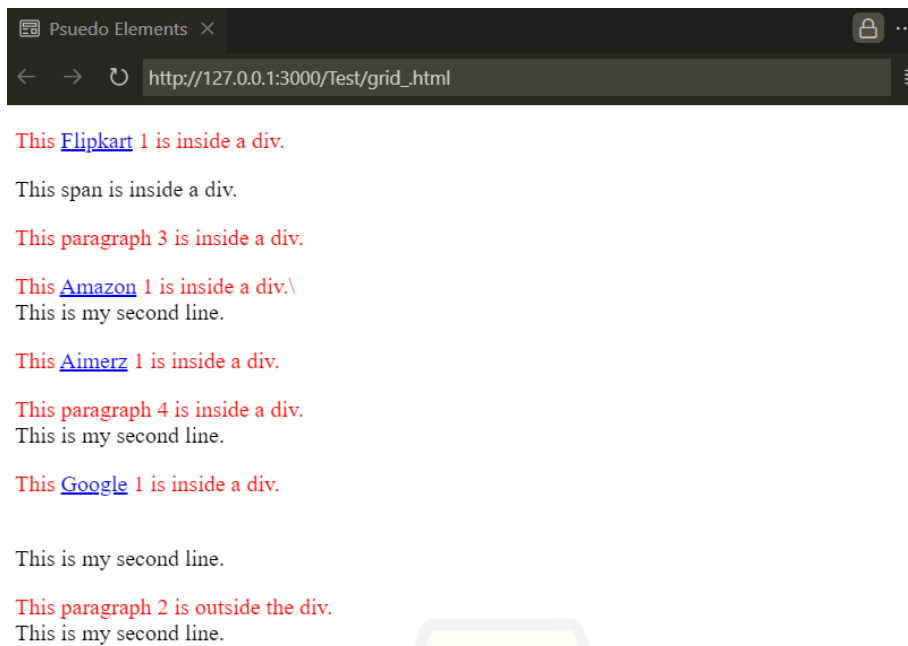
</body>
</html>
```

CSS:

Unset

```
p::first-line {
  content: " ✨ This content is added using ::after psuedo element";
  color: red;
}
```


OUTPUT:



* In the above output we can clearly see that all the `<p>` tags have inherited our new extended red colour, but are only limited till the first line for every `<p>` paragraph tag.

2.5 ::selection

The `::selection` pseudo-element in CSS is used to style the portion of an element's text that the user selects or highlights. It allows you to change the background colour, text colour, and a few other properties for selected text, providing a customized highlight effect.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Elements</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
```

```
<p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
<span>This span is inside a div.</span>
<p name="myName">This paragraph 3 is inside a div.</p>
<p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.\
  <br> This is my second line.
</p>
<p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>

<p name="personal myName">This paragraph 4 is inside a div.
  <br> This is my second line.
</p>
<p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>
<br> This is my second line.

</div>

<p>This paragraph 2 is outside the div.
  <br> This is my second line.
</p>

</body>
</html>
```

CSS:

Unset

```
::selection{
color: red;
background-color: aqua;
}
```

OUTPUT:

```

  <  →  ↻  http://127.0.0.1:3000/Test/grid_.html

  This Flipkart 1 is inside a div.
  This span is inside a div.
  This paragraph 3 is inside a div.
  This Amazon 1 is inside a div.
  This is my second line.
  This Aimerz 1 is inside a div.
  This paragraph 4 is inside a div.
  This is my second line.
  This Google 1 is inside a div.

  This is my second line.
  This paragraph 2 is outside the div.
  This is my second line.
  
```

*Whenever we hover and select the text from our browser, i.e. the pseudo element will automatically change the text colour to red and background colour to sky blue.

2.6 ::placeholder

Pseudo-classes in CSS define the special states of an element, like when it's hovered over, visited, or is the first child in its parent. They apply styles based on an element's state or behaviour, unlike pseudo-elements, which style specific parts of an element.

Let's take an example for ::placeholder pseudo elements.

HTML:

```

Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <p>This <a href="https://flipkart.com">Flipkart</a> 1 is inside a
div.</p>
      <span>This span is inside a div.</span>
      <p name="myName">This paragraph 3 is inside a div.</p>
    </div>
  </body>
</html>
  
```

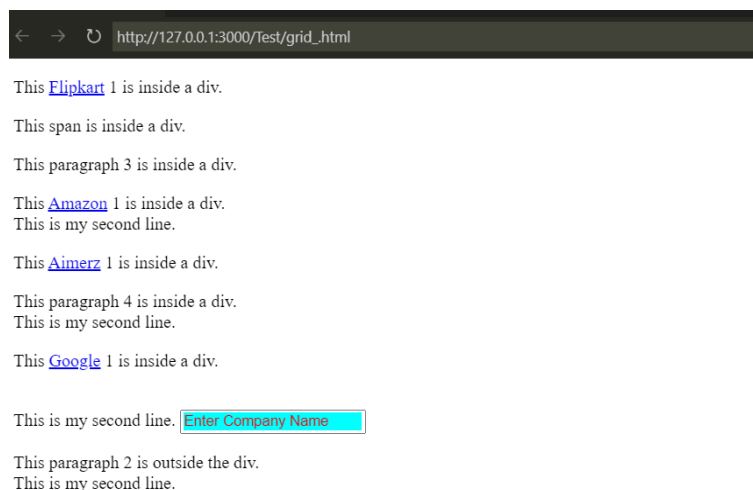
```
<p>This <a href="https://amazon.Com">Amazon</a> 1 is inside a div.  
  <br> This is my second line.  
</p>  
<p>This <a href="https://aimerz.ai">Aimerz</a> 1 is inside a div.</p>  
  
<p name="personal myName">This paragraph 4 is inside a div.  
  <br> This is my second line.  
</p>  
<p>This <a href="https://google.com">Google</a> 1 is inside a div.</p>  
<br> This is my second line.  
  
<input tyle="text" placeholder="Enter Company Name" />  
  
</div>  
  
<p>This paragraph 2 is outside the div.  
  <br> This is my second line.  
</p>  
  
</body>  
</html>
```

CSS:

Unset

```
::placeholder{  
  color: red;  
  background-color: aqua;  
}
```

OUTPUT:



Just as an example, we have added an Input tag specifically to add an placeholder “Enter Company Name” , further the background color and color properties are applied on the placeholder using ::placeholder property.

2.7 ::marker

The ::marker pseudo-element in CSS styles the bullet or number markers of list items () in ordered () or unordered () lists. It allows you to change the color, font, or size of the markers.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Elements</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <ul>
      <li> Home</li>
      <li>Students</li>
      <li>Mock Interview</li>
      <li>Assignments</li>
    </ul>
  </body>
</html>
```

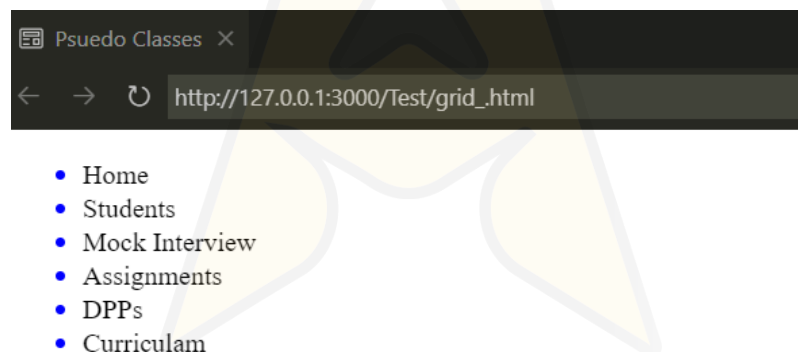
```
<li>DPPs</li>
<li>Curriculum</li>
</ul>

</body>
</html>
```

CSS:

```
Unset
ul li::marker {
  color: blue;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



3. Pseudo Classes

Pseudo-classes in CSS define the special states of an element, like when it's hovered over, visited, or is the first child in its parent. They apply styles based on an element's state or behaviour, unlike pseudo-elements, which style specific parts of an element.

Now lets, discuss on all the pseudo classes one by one with proper explanation and demonstration along with output :-

3.1 :active

Active pseudo class is used to apply the styling to an html element as soon as the user activates (clicks) on the element on which we have implemented the :active pseudo class. It acts similar to an activation button in the HTML element.

Let's take an example of :active pseudo class with output.

Example: define an :active class on the <p> tag information provided in the HTML file. Display the output for before and after the Click event.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY SAMPLE TEXT I HAVE WRITTEN</p>

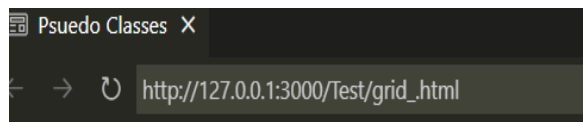
  </body>
</html>
```

CSS:

```
Unset
p:active {
  color: blue;
  background-color: yellowgreen;
  font-weight: bold;
  font-size: large;

}
```

OUTPUT:



THIS IS MY SAMPLE TEXT I HAVE WRITTEN

Fig: Before the click event



THIS IS MY SAMPLE TEXT I HAVE WRITTEN

Fig: After click event

3.2 :hover

This class is used to Select or change the appearance of the text inside the element just after when we bring the mouse over to that text.

Lets understand the concept with an example:

Example: Style the document in such a way that when we take a paragraph text and apply a background change when we hover over the text.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY SAMPLE TEXT I HAVE WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:hover {
  background-color: blue;
  font-weight: bold;
  font-size: large;
```


}

OUTPUT:

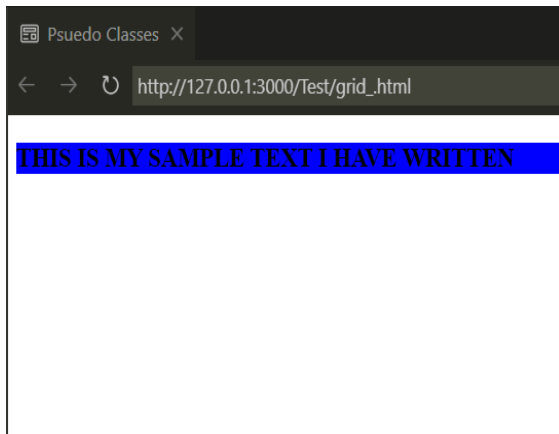


Fig: After Hovering over the text



Fig: Demonstration

3.3 :focus

In CSS, the `:focus` pseudo-class is used to style an element that has received focus, often as a result of a user interaction like tabbing through a form or clicking an input field. It is commonly applied to form elements, buttons, or links to improve accessibility and user experience by providing visual feedback when an element is in focus.

Let's take a look at `:focus` pseudo class with an example given below:-

Example: Create an input box by using Input tag, and apply `:focus` pseudo class to change the appearance of text when we are writing in the input box.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
```

```
<p>THIS IS MY SAMPLE TEXT I HAVE WRITTEN</p>
<label for="name">Name: </label><br>
<input id="name" name="name" type="text" /><br>

</body>
</html>
```

CSS:

```
Unset
input:focus {
  background-color: blue;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:

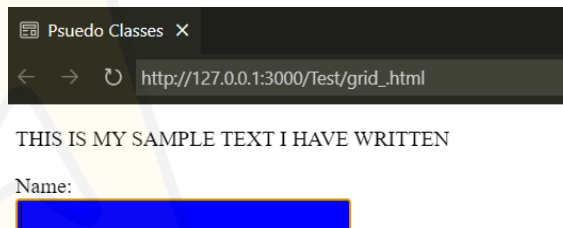
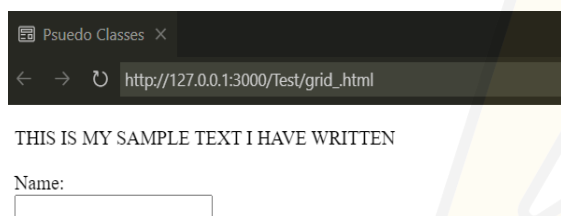


Fig: Before activation of pseudo class **Fig:** After clicking on input box(After applying prop.)

3.4 :visited

The `:visited` pseudo-class in CSS is used to style links (`<a>` elements) that the user has already clicked. It helps users tell which links they've visited, making it easier to navigate a site.

Let's implement the `:focus` pseudo class with an example given below:-

Example: Create various hyperlinks and apply `:visited` pseudo class to all the anchor `<a>` elements and change the color of all hyperlinks if they were already visited.

HTML:

Unset

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://google.com"> Google</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>

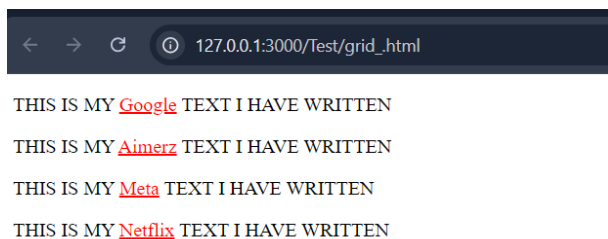
  </body>
</html>
```

CSS:

Unset

```
a:visited {
  color: red;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



3.5 :link

The `:link` pseudo-class in CSS is used to style unvisited links (`<a>` elements that haven't been clicked yet). It applies to any hyperlink that the user hasn't interacted with.

Example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://google.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
a:link{
  color: mediumseagreen;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



Fig: Before clicking any links yet

Fig: After clicking a few links

3.6 :first-child

The :first-child pseudo-class in CSS is used to select the first child element of its parent. It applies styles to the element that is the first child within a parent container.

Lets implement the :first-child pseudo class using HTML/CSS with a live example given below.

Example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

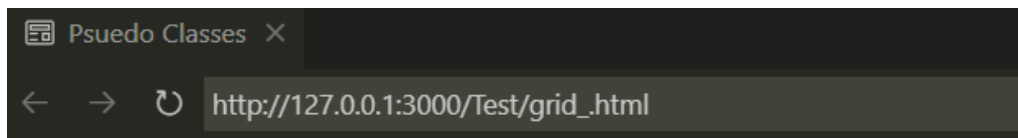
    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>

  </body>
</html>
```

CSS:

```
Unset
p:first-child{
  color: mediumseagreen;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



THIS IS MY [Google](#) TEXT I HAVE WRITTEN

THIS IS MY [Aimerz](#) TEXT I HAVE WRITTEN

THIS IS MY [Meta](#) TEXT I HAVE WRITTEN

THIS IS MY [Netflix](#) TEXT I HAVE WRITTEN

*In the above output you can clearly see that the `:first-child` property is applied to the immediate first occurrence of the tag specified.

3.7 :last-child

The `:last-child` pseudo-class in CSS is used to select the last child element within its parent container. It applies styles to the element that is the last child among all siblings.

Example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
```

```
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">
</head>
<body>
<p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE
WRITTEN</p>
<p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
<p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

<p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>
</body>
</html>
```

CSS:

```
Unset
p:last-child{
  color: mediumseagreen;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:

← → ↻ http://127.0.0.1:3000/Test/grid_html

THIS IS MY [Google](#) TEXT I HAVE WRITTEN

THIS IS MY [Aimerz](#) TEXT I HAVE WRITTEN

THIS IS MY [Meta](#) TEXT I HAVE WRITTEN

THIS IS MY [Netflix](#) TEXT I HAVE WRITTEN

3.8 :nth-child(n)

The :nth-child(n) pseudo-class in CSS is used to select elements based on their position within a parent element. The n can be a number, a keyword, or an expression, allowing for flexible selection of child elements in a sequence.

Example: Apply the colour change property on the 2nd occurrence of the paragraph <p> using the :nth-child(n) property.

HTML:

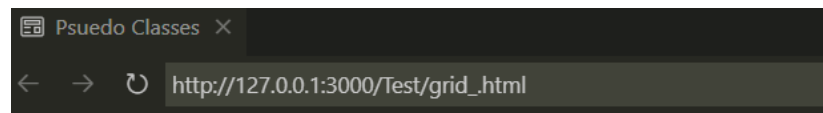
```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:nth-child(2){
  color: mediumseagreen;
  font-weight: bold;
  font-size: large;
}
```


OUTPUT:



THIS IS MY [Google](https://google.com) TEXT I HAVE WRITTEN

THIS IS MY [Aimerz](https://aimerz.ai) TEXT I HAVE WRITTEN

THIS IS MY [Meta](https://meta.com) TEXT I HAVE WRITTEN

THIS IS MY [Netflix](https://netflix.com) TEXT I HAVE WRITTEN

3.9 :nth-last-child(n)

The :nth-last-child(n) pseudo-class in CSS is used to select elements based on their position within their parent, counting from the last child to the first. It works similarly to :nth-child(n), but the count starts from the end rather than the beginning.

Example: Apply the colour change property on the 2nd occurrence of the paragraph <p> starting from last using the :nth-child(n) property.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>
```

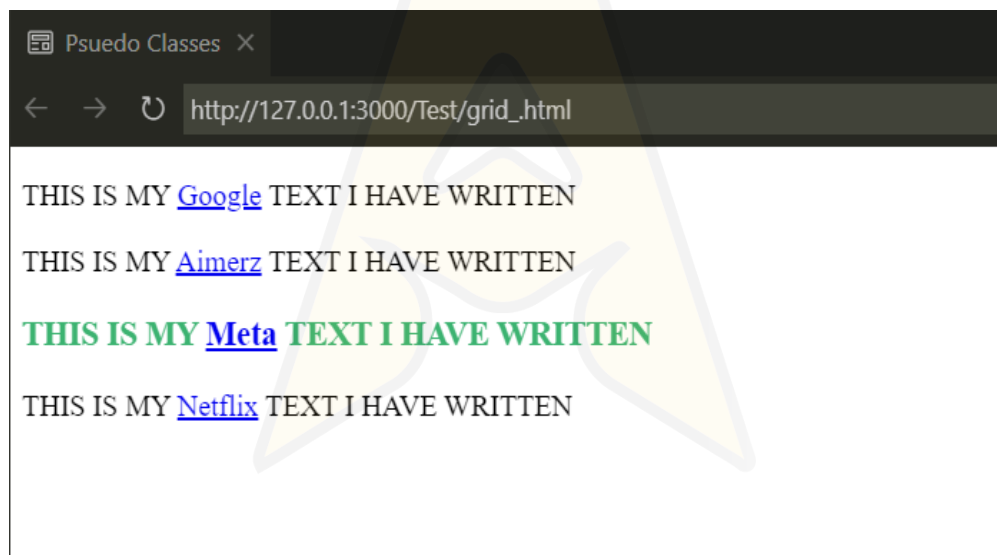
```
<p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>
```

```
</body>  
</html>
```

CSS:

```
Unset  
p:nth-last-child(2){  
  color: mediumseagreen;  
  font-weight: bold;  
  font-size: large;  
}
```

OUTPUT:



3.10 :nth-of-type(n)

The `:nth-of-type(n)` pseudo-class in CSS is used to select elements based on their position among siblings of the same type (tag), regardless of other elements present. It allows you to style specific elements of a certain type within a parent container.

Example:

HTML:

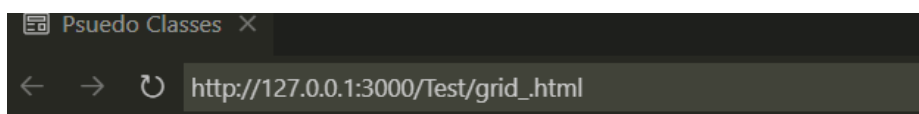
```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:nth-of-type(2n){
  color: mediumseagreen;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



THIS IS MY [Google](https://googl.com) TEXT I HAVE WRITTEN

THIS IS MY [Aimerz](https://aimerz.ai) TEXT I HAVE WRITTEN

THIS IS MY [Meta](https://meta.com) TEXT I HAVE WRITTEN

THIS IS MY [Netflix](https://netflix.com) TEXT I HAVE WRITTEN

3.11 :nth-last-of-type(n)

The :nth-last-of-type(n) pseudo-class in CSS selects elements of the same type, starting from the last. It lets you style specific elements based on their position from the end of their parent.

Example:

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>

    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:nth-last-of-type(2){
  color: greenyellow;
  font-weight: bold;
  font-size: large;
}
```

OUTPUT:



3.12 :not(selector)

The `:not(selector)` pseudo-class in CSS is used to select elements that do not match a specific selector. It allows you to apply styles to elements that don't fit a particular condition, providing more control over your styles.

Example:

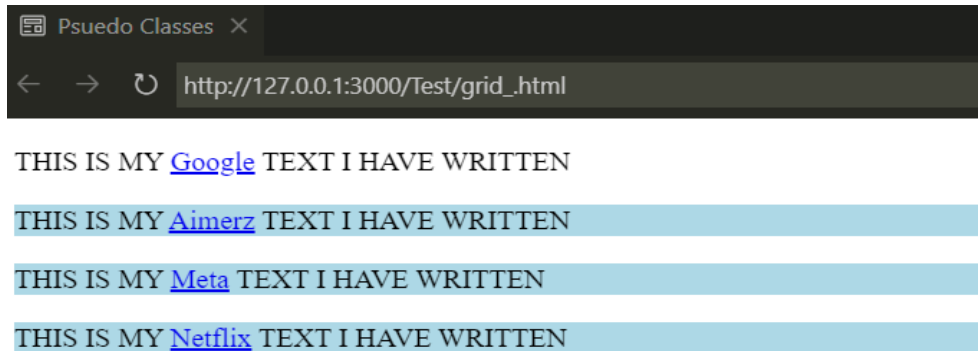
HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>
    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:not(:first-child) {
  background-color: lightblue;
}
```

OUTPUT:



3.13 :first-of-type

The `:first-of-type` pseudo-class in CSS is used to select the first element of a given type within its parent. It applies styles to the first occurrence of a specific tag (like the first `<p>`, ``, or `<div>`) among its siblings.

Example:

HTML:

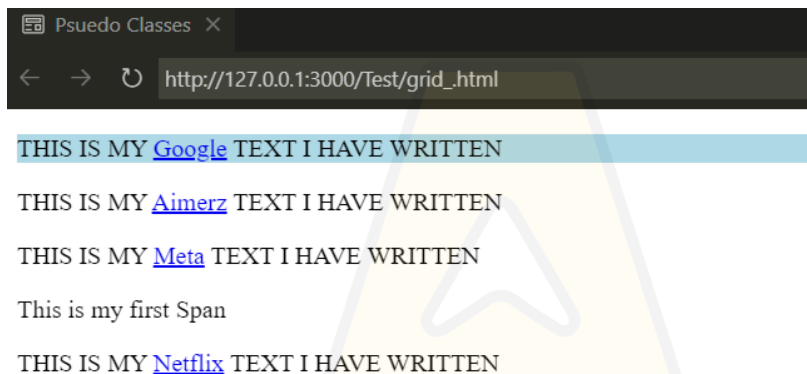
```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>
```

```
<span> This is my first Span</span>
<p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>
</body>
</html>
```

CSS:

```
Unset
p:first-of-type {
  background-color: lightblue;
}
```

OUTPUT:



3.14 :last-of-type

The `:last-of-type` pseudo-class in CSS selects the last element of a specific type within its parent. It helps you style the last instance of a particular tag (e.g., the last `<p>`, last ``, etc.), regardless of other elements present.

Example:

HTML:

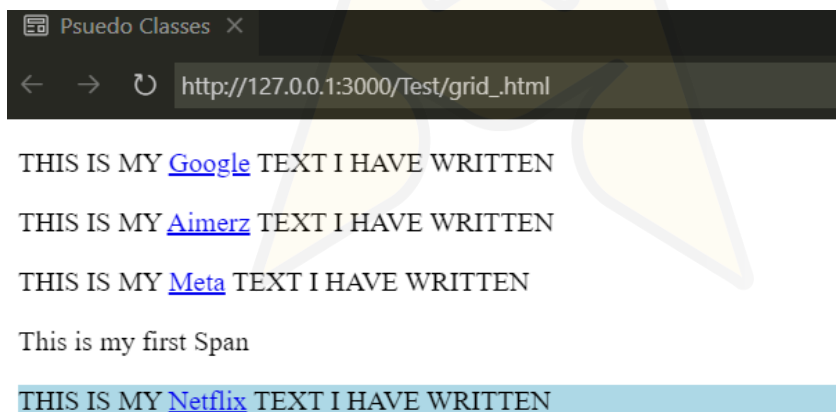
```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="style11.css">
</head>
<body>
<p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE
WRITTEN</p>
<p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
<p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>
  <span> This is my first Span</span>
<p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>
</body>
</html>
```

CSS:

```
Unset
p:last-of-type {
  background-color: lightblue;
}
```

OUTPUT:



3.15 :empty

The :empty pseudo-class in CSS selects elements that do not have any children. This includes elements with no text, no elements, or no whitespace inside them.

Example: Use the :empty pseudo class and improve the User Interface by adding a border if text inside any element tag is null.

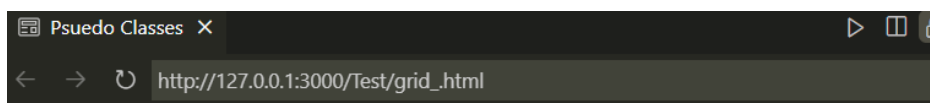
HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p>THIS IS MY <a href="https://googl.com"> Google</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://aimerz.ai"> Aimerz</a> TEXT I HAVE
WRITTEN</p>
    <p>THIS IS MY <a href="https://meta.com"> Meta</a> TEXT I HAVE WRITTEN</p>
    <span> This is my first Span</span>
    <p>THIS IS MY <a href="https://netflix.com"> Netflix</a> TEXT I HAVE
WRITTEN</p>
  </body>
</html>
```

CSS:

```
Unset
p:empty {
  border: 2px dashed red;
}
```

OUTPUT:



THIS IS MY [Google](https://googl.com) TEXT I HAVE WRITTEN

THIS IS MY [Aimerz](https://aimerz.ai) TEXT I HAVE WRITTEN

THIS IS MY [Meta](https://meta.com) TEXT I HAVE WRITTEN

This is my first Span

THIS IS MY [Netflix](https://netflix.com) TEXT I HAVE WRITTEN

3.16 :checked

The `:checked` pseudo-class in CSS is used to select and style elements that are in a "checked" state. This typically applies to form elements like checkboxes, radio buttons, and `<option>` elements in a dropdown when they are selected or checked.

Example: Create a input radio button and further style in such a way that the checked button has a red color appearance.

HTML:

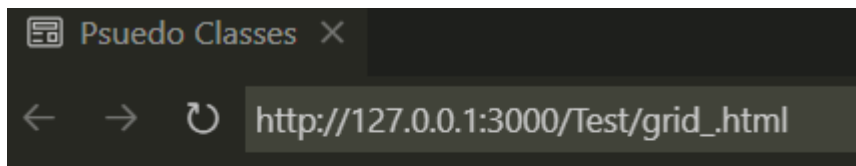
```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>

    <label><input type="radio" name="option" checked> Option 1 </label>
    <label><input type="radio" name="option"> Option Second</label>
  </body>
</html>
```

CSS:

```
Unset
input:checked {
  accent-color: brown;
  text-decoration: aqua
}
```

OUTPUT:

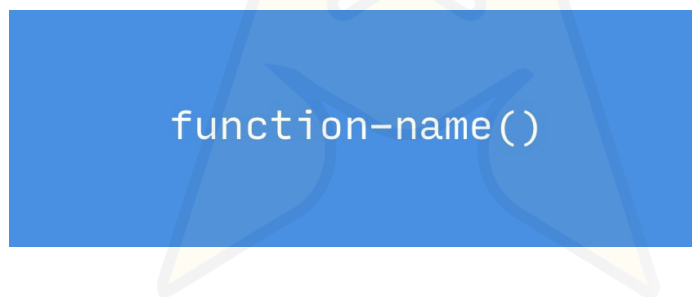


☐ Option 1 ☒ Option Second

4. More CSS Functions

Introduction

CSS functions are built-in functions that allow you to perform calculations or manipulate values in your styles. They provide flexibility and dynamic behaviour when styling elements.



4.2 attr()

The attr() function in CSS is used to retrieve and display the value of an HTML attribute. It's mostly used with the content property, typically in combination with the ::before or ::after pseudo-elements, to insert content dynamically based on an element's attribute.

```
Syntax :    attr(attribute_name);  
Example :  p::before {  
              content: attr(data-tooltip);  
            }
```

Key Benefits:

- Dynamic, attribute-driven styling
- Reduces redundancy
- Enhances flexibility and adaptability of content styli

Example: Show all the anchor tag hyperlinks references in a bracket after the element text.

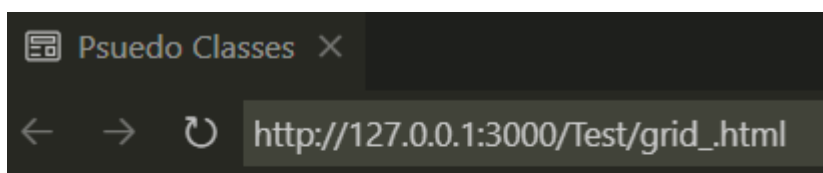
HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <p><a href="https://google.com"> Google Web Search .</a></p>
    <a href="https://aimerz.ai"> Aimerz Pathway Website .</a>
  </body>
</html>
```

CSS:

```
Unset
a:after {content: " { " attr(href) " } "};
```

OUTPUT:



Google Web Search . { https://google.com }.

Aimerz Pathway Website . { https://aimerz.ai }.

4.3 calc()

The `calc()` function in CSS allows you to perform calculations to determine CSS property values. It lets you combine different units (such as percentages, pixels, ems, etc.) or use mathematical operations to create more dynamic and responsive layouts.

```
Syntax :    calc(expression) ;  
Example :    div {  
                width: calc(100% - 40px) ;  
            }
```

Key Benefits of the `calc()` Function:

Flexible Layouts: Combines different units for adaptable designs.

Dynamic Calculations: Adjusts sizes based on other properties directly in CSS.

Unit Mixing: Seamlessly mixes units (e.g., percentages and pixels).

Responsive Design: Integrates viewport units for fluid layouts.

Simplifies Complexity: Handles intricate sizing without extra rules.

Example: Create a div containing various list items and use `calc()` function to resize in CSS.

HTML:

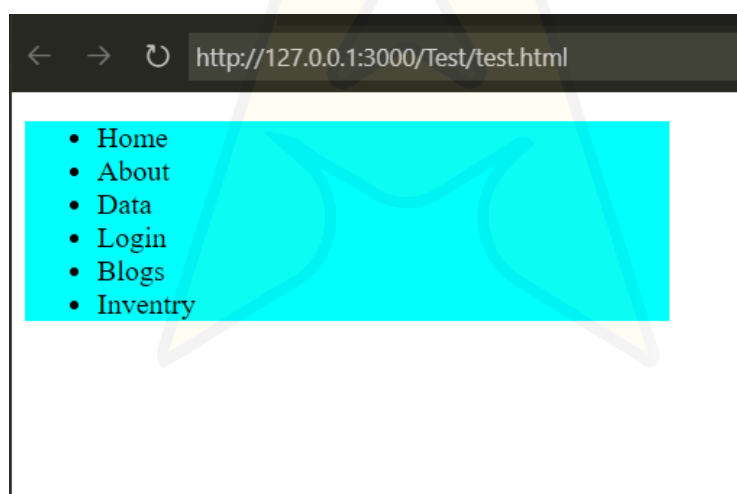
```
Unset  
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <title>Psuedo Classes</title>  
    <meta name="description" content="">  
    <meta name="viewport" content="width=device-width, initial-scale=1">  
    <link rel="stylesheet" href="style11.css">  
  </head>  
  <body>  
    <div>
```

```
<ul>
  <li>Home </li>
  <li>About</li>
  <li>Data</li>
  <li>Login</li>
  <li>Blogs</li>
  <li>Inventory</li>
</ul>
</div>
</body>
</html>
```

CSS:

```
Unset
div {
  width: calc(100% - 100px);
  background-color: aqua; /* 100% of the parent width minus 50px */
}
```

OUTPUT:



4.4 max()

The `max()` function in CSS lets you choose the largest value from the options you provide. It helps create flexible layouts by ensuring that a property's value doesn't go below a certain size, making designs more adaptable to different screen sizes.

Example: By using the max() function specify the maximum width of the div defined and also provide the output for it.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <ul>
        <li>Home </li>
        <li>About</li>
        <li>Data</li>
        <li>Login</li>
        <li>Blogs</li>
        <li>Inventry</li>
      </ul>
    </div>
  </body>
</html>
```

CSS:

```
Unset
div {

  width: max(500px);
  background-color: aqua; /* 100% of the parent width minus 50px */
}
```

OUTPUT:



4.5 min()

The min() function in CSS allows you to set a value by picking the smallest of the provided values. It's useful for creating flexible designs that adapt to different screen sizes, ensuring that a property's value won't exceed a certain limit.

Example: By using the min() function specify the minimum width of the div defined and also provide the output for it.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <ul>
        <li>Home </li>
        <li>About</li>
        <li>Data</li>
        <li>Login</li>
        <li>Blogs</li>
        <li>Inventory</li>
      </ul>
    </div>
  </body>
</html>
```

CSS:

```
Unset
div {

  width: min(800px);
  background-color: aqua; /* 100% of the parent width minus 50px */
}
```


OUTPUT:



4.6 var()

The var() function in CSS is used to access and use custom CSS variables. It lets you reuse values throughout your stylesheet, making it easier to manage and update your CSS.

Use Cases:

- **Theming:** Easily switch between dark and light themes by updating variables.
- **Consistent Spacing/Styling:** Use the same padding, margin, or font size across multiple elements.
- **Dynamic Values:** Adjust variables based on media queries or other conditions to make the design responsive.
- **The var()** function is an essential tool in modern CSS, allowing implementation to make implementation more dynamic and reusable.

Example: By using the var() function specify the background colour of the div defined and also provide the output for it.

HTML:

```
Unset
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Psuedo Classes</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="style11.css">
  </head>
  <body>
    <div>
      <ul>
        <li>Home </li>
        <li>About</li>
        <li>Data</li>
```

```
<li>Login</li>
<li>Blogs</li>
<li>Inventory</li>
</ul>
</div>

</body>
</html>
```

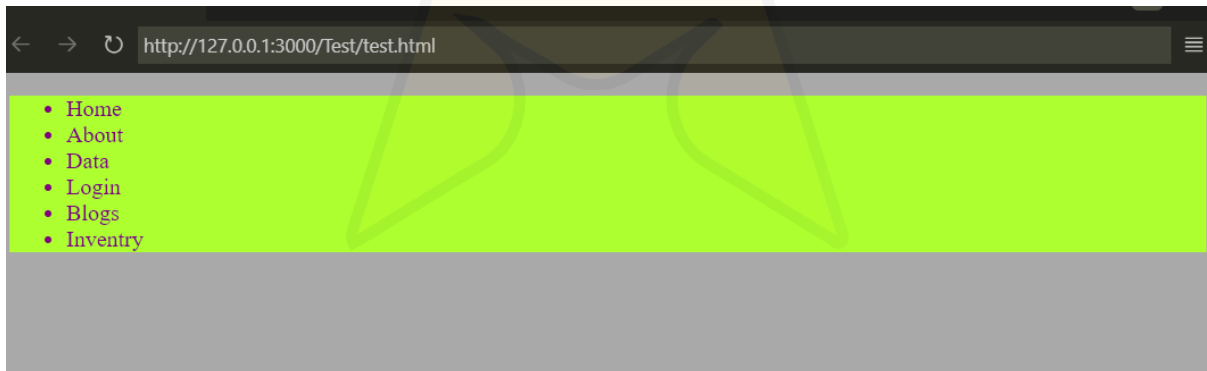
CSS:

```
Unset
:root {
  color: purple;
  --background-color: greenyellow;
  background-color: darkgray;
}

div{

  background-color: var(--background-color);
}
```

OUTPUT:



THANK YOU

