Part 1: CSS Positioning

Objective: Create a web page demonstrating different CSS positioning techniques.
Instructions:
Create an HTML file named index.html.
Add a div element with the class container and three child div elements with classes absolute, relative, and fixed.
Style the container to have a width of 500px and height of 300px.
Apply different positioning styles to each child div?

Answer:-

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CSS Positioning</title>
    <style>
        .container {
            width: 500px;
            height: 300px;
            border: 1px solid black;
            position: relative;
        }

        .absolute {
            position: absolute;
            top: 20px;
            left: 20px;
            width: 100px;
            height: 100px;
            background-color: lightblue;
        }

        .relative {
            position: relative;
            top: 40px;
            left: 40px;
            width: 100px;
            height: 100px;
            background-color: lightgreen;
```

```
        }

        .fixed {
            position: fixed;
            bottom: 20px;
            right: 20px;
            width: 100px;
            height: 100px;
            background-color: lightcoral;
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="absolute">Absolute</div>
        <div class="relative">Relative</div>
        <div class="fixed">Fixed</div>
    </div>
</body>
</html>
```

Output:-

Part 2

Try changing the width and give only 10px to border property. Mention what changes you have noticed with the content. Hint: Create a html with div containers and classes accordingly.

```
.border-box, .content-box {
width: 200px;
height: 100px;
margin: 20px;
padding: 20px;
border: 10px solid black;
}

.border-box {
box-sizing: border-box;
background-color: lightyellow;
}

.content-box {
box-sizing: content-box;
background-color: lightgray;
}
```

Answer:-

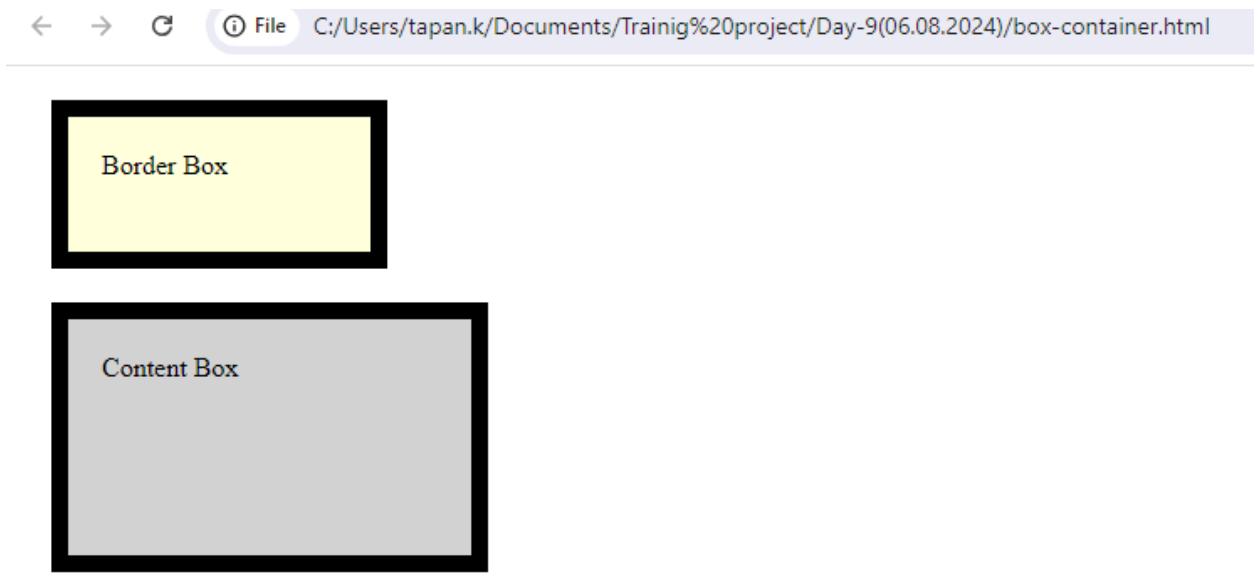
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Box Sizing</title>
    <style>
        .border-box, .content-box {
            width: 200px;
            height: 100px;
            margin: 20px;
            padding: 20px;
            border: 10px solid black;
        }

        .border-box {
            box-sizing: border-box;
```

```
            background-color: lightyellow;
        }

        .content-box {
            box-sizing: content-box;
            background-color: lightgray;
        }
    </style>
</head>
<body>
    <div class="border-box">Border Box</div>
    <div class="content-box">Content Box</div>
</body>
</html>
```

Output:-



Part 3

Javascript – show difference between substr and substring with negative index and positive index for the string “The world is wonderful”?

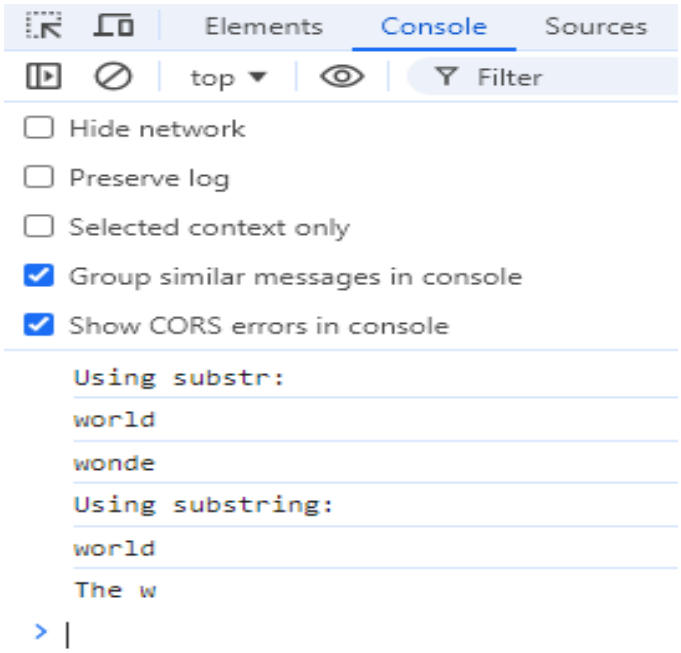
```
<!DOCTYPE html>
```

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Substr vs Substring</title>
</head>
<body>
    <script>
        const str = "The world is wonderful";

        console.log("Using substr:");
        console.log(str.substr(4, 5)); // "world"
        console.log(str.substr(-9, 5)); // "wonde" (counting from the end)

        console.log("Using substring:");
        console.log(str.substring(4, 9)); // "world"
        console.log(str.substring(-9, 5)); // "The w" (negative index is
treated as 0)
    </script>
</body>
</html>
```

Output:-



Part 4

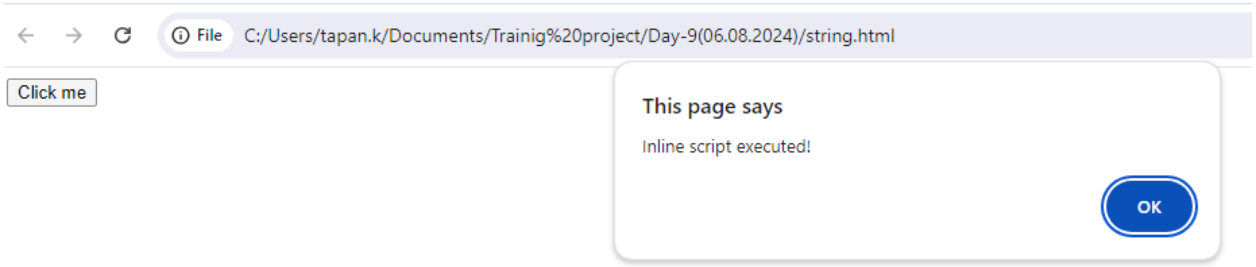
Javascript : Show what's inline, internal and external scripts?

Answer:-.

Inline Script:-

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inline Script</title>
</head>
<body>
    <button onclick="alert('Inline script executed!')">Click me</button>
</body>
</html>
```

Output:-

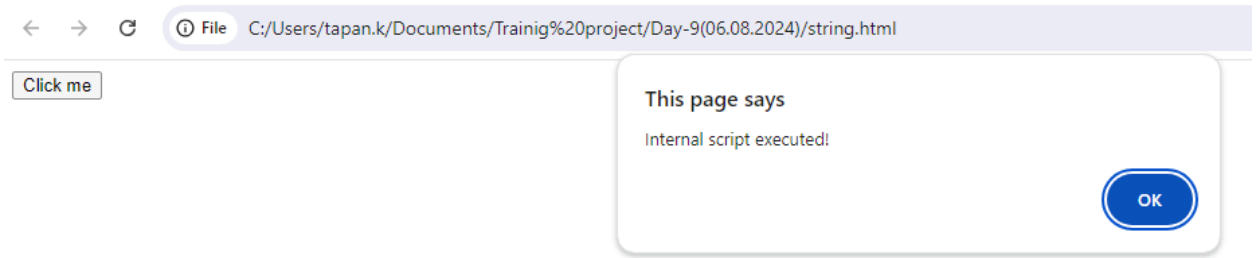


Internal Script

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <title>Internal Script</title>
    <script>
        function showMessage() {
            alert('Internal script executed!');
        }
    </script>
</head>
<body>
    <button onclick="showMessage()">Click me</button>
</body>
```

Output:-



External Script

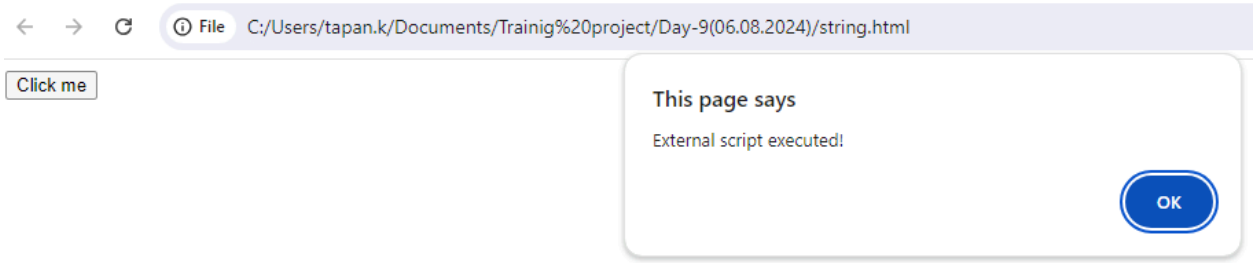
Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>External Script</title>
    <script src="script.js" defer></script>
</head>
<body>
    <button onclick="showMessage()">Click me</button>
</body>
</html>
```

External Script:

```
function showMessage() {
    alert('External script executed!');
}
```

Output:-



Part 5:

Javascript: As per naming convention, which variable is advisable to use for functions or arrays: const or let or var?

For functions and arrays, it's advisable to use const when you don't expect to reassign the variable, as it prevents reassignment and makes the code easier to understand and maintain. Use let if you need to reassign the variable.Avoid using var as it has function scope and can lead to bugs due to its hoisting behavior.

Example with const

```
const myArray = [1, 2, 3];
const myFunction = function() {
    console.log("Hello, world!");
};

myArray.push(4); // Allowed
myFunction(); // Executes the function
```

Example with let

```
let myArray = [1, 2, 3];
let myFunction = function() {
    console.log("Hello, world!");
};

myArray = [4, 5, 6]; // Reassigning the array
myFunction = function() {
    console.log("Goodbye, world!");
};

myArray.push(7); // Allowed
myFunction(); // Executes the new function
```