Exercise 1: JavaScript Debugging

The corrected code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Toggle Element</title>

</head>

<body>

    <button onclick="toggleElement()">Toggle Element</button>

    <div id="target" style="display: none;">This is the target element.</div>

    <script>

**function toggleElement() {**

**var element = document.getElementById("target");**

**var style = window.getComputedStyle(element);**

**var display = style.getPropertyValue("display");**

**if (display === "none") {**

**element.style.display = "block";**

**} else {**

**element.style.display = "none";**

**}**

**}**

    </script>

</body>

</html>

The error was in the line:

  element.style.display = (element.style.display === "none") ? "block" : "none";

in the toggleElement function

When you set an element's display property directly via JavaScript, it's accessing the inline style of the element. However, if the style is defined in an external CSS file or in a <style> block in the <head> section, element.style.display will be an empty string. Therefore, the condition (element.style.display === "none") will always evaluate to false, and the code will always set the display property to "none".

To fix this, we can use window.getComputedStyle() to get the computed style of the element.

In this corrected code, window.getComputedStyle(element) retrieves the computed style of the element, and style.getPropertyValue("display") gets the value of the display property. The if statement then correctly toggles the visibility of the element based on its computed style. Now, clicking the button will toggle the visibility of the targeted element properly.

Exercise 2: CSS Troubleshooting

The corrected code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Centered Container</title>

<style>

**.container {**

**position: absolute;**

**top: 50%;**

**left: 50%;**

**transform: translate(-50%, -50%);**

**width: 50%;**

**background-color: #f0f0f0;**

**padding: 20px;**

**}**

</style>

</head>

<body>

<div class="container">

<h1>Centered Container</h1>

<p>This container should be centered on the page.</p>

</div>

</body>

</html>

The original CSS code centers the container horizontally on the page using

margin: auto;

However, it doesn't center the container vertically on the page.

In the corrected code:

position: absolute; is used to take the element out of the normal document flow.

top: 50%; and left: 50%; position the element 50% from the top and 50% from the left of the viewport.

transform: translate(-50%, -50%); moves the container back by 50% of its own width and height, effectively centering it both horizontally and vertically relative to its containing element (in this case, the viewport).

With these changes, the container will be both horizontally and vertically centered on the page

Exercise 3: Debugging JavaScript Functions

function calculateSum(arr) {

let sum = 0;

for (let i = 0; i < arr.length; i++) {

sum += arr[i];

}

return sum;

}

const numbers = [1, 2, 3, 4, 5];

const result = calculateSum(numbers);

console.log(result); // Should output 15

There are no errors in the above code. It defines a function called calculateSum which takes an array of numbers as input and calculates their sum using a for loop. Then, it creates an array called numbers containing [1, 2, 3, 4, 5] and calls the calculateSum function with this array. The expected output is 15, and it will be displayed in the console when you run this code. The console.log(result) statement is correctly used to output the result of the calculation to the console.

Exercise 2: Debugging CSS Styling Issue

The corrected code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Styling Debugging Exercise</title>

<style>

**.container {**

**width: 100%; /\* 100% width to fill the parent element \*/**

**display: flex; /\* flexbox for layout \*/**

**justify-content: space-around; /\* Center the boxes with space around them \*/**

**background-color: #f0f0f0;**

**padding: 20px;**

**box-sizing: border-box; /\* Include padding in the total width \*/**

**}**

**.box {**

**width: 100px;**

**height: 100px;**

**background-color: #007bff;**

**color: #ffffff;**

**text-align: center;**

**line-height: 100px;**

**}**

</style>

</head>

<body>

<div class="container">

<div class="box">Box 1</div>

<div class="box">Box 2</div>

<div class="box">Box 3</div>

</div>

</body>

</html>

Changes made:

Container CSS:

Set width to 100% to make the container fill the parent element.

Used display: flex; to enable flexbox layout for the container.

Added justify-content: space-around; to center the boxes with space around them.

Added box-sizing: border-box; to include padding in the total width of the container.

Box CSS:

No changes were made to the box styles, as they are already defining the box appearance correctly.

With these changes, the boxes will be centered horizontally inside the container with space around them. The padding inside the container is also included in the total width.