Web Debug Solutions:

Exercise 1:

Given Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Debugging Exercise</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f4f4f4;
color: #333;
margin: 20px;
}

h1 {
color: #007bff;
}

p {
font-size: 16px;
margin-bottom: 20px;
}

.important-text {
font-weight: bold;
color: #d9534f;
}
</style>
</head>
<body>

<h1>Debugging Exercise</h1>

<p>This is a paragraph with some <span class="important-text">important text</span>.</p>
```

Here's an unordered list:

Item 1
Item 2
Item 3

And here's an ordered list:

First item
Second item
Third item

This is a link to example.com.

<script>
console.log("Debugging exercise script");
</script>

</body>
</html>

Corrected Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Debugging Exercise</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f4f4f4;
color: #333;
margin: 20px;
}

h1 {
color: #007bff;
}
```

```
p {
font-size: 16px;
margin-bottom: 20px;
}

.important-text {
font-weight: 700;
color: #d9534f;
}
</style>
</head>
<body>

<h1>Debugging Exercise</h1>

<p>This is a paragraph with some <span class="important-text">important
text</span>.</p>

<p>Here's an unordered list:</p>
<ul>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>

<p>And here's an ordered list:</p>
<ol>
<li>First item</li>
<li>Second item</li>
<li>Third item</li>
</ol>

<p>This is a <a href="https://www.example.com">link to example.com</a>.</p>

<script>
console.log("Debugging exercise script");
```

</script>

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

Explanation:

The issue is in the important-text class definition within the <style> section. The issue is with the declaration of the font weight. Instead of "bold", it should be "700" or any other numerical value.

Exercise 2:

Given 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");
            element.style.display = (element.style.display === "none") ? "block" : "none";
        }
    </script>
</body>
</html>
```

Explanation:

This code will work as expected in most cases. However, if the display property of the element is set in the CSS file or inline, and it's not explicitly set to "none" initially, then element.style.display will return an empty string initially instead of "none". This can happen if the element is initially visible, but hidden by a CSS rule.

To fix the issue, we can check the computed style instead of relying solely on the inline style.

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 computedStyle = window.getComputedStyle(element); // Get the computed style
element.style.display = (computedStyle.display === "none") ? "block" : "none";
}


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

Exercise 3:

Given 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 {
margin: auto;
width: 50%;
background-color: #f0f0f0;
padding: 20px;
}
</style>
</head>
```

<body>

<h1>Centered Container</h1>
This container should be centered on the page.

</body>
</html>

Explanation:

The CSS code provided seems correct for centering the container horizontally using margin auto. The issue might be related to the width percentage or the parent container's size. However, to ensure proper centering, we should also specify a height for the container.

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 {
margin: auto;
width: 50%;
height: 200px; /* Added height for demonstration */
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>
```

Exercise 4:

Given Code:

```
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
```

Explanation:

The function seems logically correct, but it lacks error handling, which might cause issues if the input arr is not an array or if its elements are not numeric.
To make the function more robust, we can add error handling to check if the input is an array and if its elements are numeric before summing them up.

Corrected Code:

This revised function will throw an error if the input is not an array or if any of its elements are not numeric, ensuring more robustness and preventing unexpected behavior.

```
function calculateSum(arr) {

if (!Array.isArray(arr)) {

throw new Error("Input is not an array");

}



let sum = 0;
```

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

if (typeof arr[i] !== 'number' || isNaN(arr[i])) {

throw new Error("Array contains non-numeric value");

}

sum += arr[i];

}

return sum;

}



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

const result = calculateSum(numbers);

console.log(result);
```

Exercise 5:

Given 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: 50%;
margin: 0 auto;
background-color: #f0f0f0;
padding: 20px;
```

}

.box {
width: 100px;
height: 100px;
background-color: #007bff;
color: #ffffff;
text-align: center;
line-height: 100px;
}
</style>
</head>
<body>

Box 1
Box 2
Box 3

</body>
</html>

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%; /* Set width to 100% to occupy the entire width */
display: flex; /* Use flexbox for horizontal alignment */
justify-content: center; /* Center the flex items horizontally */
background-color: #f0f0f0;
padding: 20px;
}

.box {
width: 100px;
height: 100px;
```

```
background-color: #007bff;
color: #ffffff;
text-align: center;
line-height: 100px;
margin: 0 10px; /* Add margin to separate the boxes */
}

</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>
```