Week-5 Assignment

1) Exercise: The task is to identify and fix the errors in the 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>

<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>
```

Instructions for debugging:

Identify and fix the errors in the HTML and CSS code.
Pay attention to missing tags, incorrect attribute values, and CSS styles.

Correct 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>
/* Error: Missing colons after property names */
/* Fix: Add colons to separate property names from values */
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>

<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>

</style>

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

</body>
</html>
```

Explanation of the errors and fixes:

1)Missing colons in CSS properties:

CSS property names and values must be separated by colons.

The original code was missing colons after font-family, background-color, color, font-size, margin-bottom, font-weight, and color.

The fix adds the missing colons to make the CSS rules valid.

2)Missing closing </style> tag:

HTML elements must be properly closed to ensure valid structure.

The <style> element was missing its closing tag, which could lead to parsing errors.

The fix adds </style> before the <script> tag to close the style element correctly.

2) Exercise : JavaScript Debugging

Problem Statement:

You've been given a simple JavaScript code snippet that's intended to toggle the visibility of an element when a button is clicked. However, it's not working as expected.

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>
```

Tasks:

Identify the issue in the provided JavaScript code.
Debug and fix the code so that clicking the button toggles the visibility of the element.

Correct 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");

      // Error: Incorrect comparison due to extra parentheses
      // Fix: Remove unnecessary parentheses
      element.style.display = element.style.display === "none" ? "block" : "none";
    }
  </script>
</body>
</html>
```

Explanation:

- Incorrect comparison:
    - The original code had (element.style.display === "none") within parentheses.
    - This caused the comparison to evaluate as a boolean value (true or false) first, rather than comparing the actual strings.
    - The fix removes the extra parentheses, ensuring the correct string comparison.
- Unnecessary conditional statement:
    - While not strictly an error, the original code used a ternary operator to conditionally set the display property.
    - This can be simplified to a direct assignment without affecting functionality.
    - The corrected code directly assigns "block" or "none" based on the current value, making the logic more concise.

3) Exercise : CSS Troubleshooting

Problem Statement:

You've been given an HTML and CSS code snippet that's supposed to create a centered, responsive container. However, it's not displaying as expected.

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>
<div class="container">
<h1>Centered Container</h1>
<p>This container should be centered on the page.</p>
</div>
</body>
</html>
```

Correct 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>
/* Error: Missing display property for horizontal centering */
/* Fix: Add "display: block" to allow margin: auto to work */
.container {
  display: block;  /* Added to enable horizontal centering */
  margin: auto;
  width: 50%;
  background-color: #f0f0f0;
  padding: 20px;
}
</style>
</head>
<body>
<div class="container">
<h1>Centered Container</h1>
<p>This container should now be centered on the page.</p>
</div>
</body>
</html>
```

Explanation:

- Missing display property:
    - The margin: auto property only centers block-level elements horizontally.
    - The .container element was missing a display property to define its behavior.
    - The fix adds display: block to make it a block-level element, enabling horizontal centering.

4) Exercise : Debugging JavaScript Functions
Objective: Identify and fix issues in JavaScript functions.

This code snippet with a JavaScript function that performs a specific task, but contains bugs or inefficiencies.
Debug the function and ensure it works correctly and efficiently.

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
```

Correct code:

```
// Error: Missing semicolon after the function declaration
// Fix: Add semicolon to avoid potential syntax errors
function calculateSum(arr) {
  let sum = 0;

  // Error: Inconsistent indentation within the loop
  // Fix: Indent the loop body consistently for readability
  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 of the errors:

1. Missing semicolon:
    - JavaScript generally requires semicolons to terminate statements.
    - The function declaration function calculateSum(arr) { was missing a semicolon, which could lead to potential syntax errors in certain cases.
2. Inconsistent indentation:
    - While indentation doesn't directly affect code execution, it's essential for readability and maintainability.
    - The loop body was not indented consistently with the rest of the function, making it harder to visually parse the code's structure.

5) Exercise 2: Debugging CSS Styling Issues
Objective: Identify and fix CSS styling issues to achieve the desired layout.

This code snippet with HTML and CSS code that creates a specific layout, but contains CSS issues like misalignment, overlapping elements, or incorrect colors.
Debug the CSS to achieve the desired layout.

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>
```

```
<div class="container">
<div class="box">Box 1</div>
<div class="box">Box 2</div>
<div class="box">Box 3</div>
</div>
</body>
</html>
```

Correct 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>
/* Error: Missing display property for box elements
/* Fix: Add "display: inline-block" to align boxes horizontally */
.box {
  display: inline-block; /* Added to align boxes horizontally */
  width: 100px;
  height: 100px;
  background-color: #007bff;
  color: #ffffff;
  text-align: center;
  line-height: 100px;
  margin: 10px; /* Added to space out the boxes */
}

.container {
  width: 50%;
  margin: 0 auto;
  background-color: #f0f0f0;
  padding: 20px;
}
</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>
```

Explanation of the errors and fixes:

1. Missing display property for box elements:
    - The .box elements were missing a display property, causing them to stack vertically by default.
    - The fix adds display: inline-block to make them inline-level elements, aligning them horizontally within the container.
2. Overlapping boxes (optional fix):
    - While not strictly an error, the boxes were directly adjacent to each other.
    - The optional fix adds margin: 10px to each box to create some spacing between them.