

Javascript5-Day 04-Hands_On_Problem

Problem 1:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">


  <!-- Requirement 1: Viewport meta tag for mobile responsiveness -->

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

  <title>Responsive Web Page - Problem 1</title>

  <style>

    /* Base styling for desktop view */

    body {

      margin: 0;

      font-family: Arial, sans-serif;

      background-color: #f4f7fb; /* Desktop background color */

    }

    header {

      background: #2c3e50;

      color: white;

      padding: 15px;

      text-align: center;

    }

    /* Desktop navigation (horizontal layout) */

    nav {

      display: flex; /* Desktop navigation in row */

      justify-content: center;

      background: #34495e;

    }

    nav a {
```

```

    color: white;

    text-decoration: none;

    padding: 15px 20px;

    display: block;
}
nav a:hover {
    background: #1abc9c;
}
main {
    padding: 20px;
}
h1 {
    font-size: 32px; /* Desktop font size */
}
p {
    font-size: 18px; /* Desktop font size */
}

/*
    Requirement 2, 3, 4: Media Queries for Mobile View
*/
/* This media query runs when screen width is 600px or less (Mobile) */
@media (max-width: 600px) {
    /* Requirement 2: Change background color on mobile */
    body {
        background-color: #fff3cd; /* Mobile background color */
    }

    /* Requirement 3: Adjust font size for smaller screens */
    h1 {
        font-size: 24px; /* Smaller heading font for mobile */
    }
}

```

```

    p {
        font-size: 16px; /* Smaller paragraph font for mobile */
    }

    /* Requirement 4: Convert navigation into vertical layout on mobile */
    nav {
        flex-direction: column; /* Navigation becomes vertical */
        align-items: center;
    }
    nav a {
        width: 100%;
        text-align: center;
        border-bottom: 1px solid #ffffff33;
    }
}
</style>
</head>
<body>
    <header>
        <h1>Responsive Website</h1>
        <p>This page adapts to mobile and desktop screens.</p>
    </header>
    <!-- Navigation Menu -->
    <nav>
        <a href="#">Home</a>
        <a href="#">Products</a>
        <a href="#">Contact</a>
    </nav>
    <main>
        <h2>About This Page</h2>
        <p>

```

Resize the browser window or open developer tools and switch to mobile view.

You will see background color change, text size reduce, and menu become vertical.

</p>

</main>

</body>

</html>

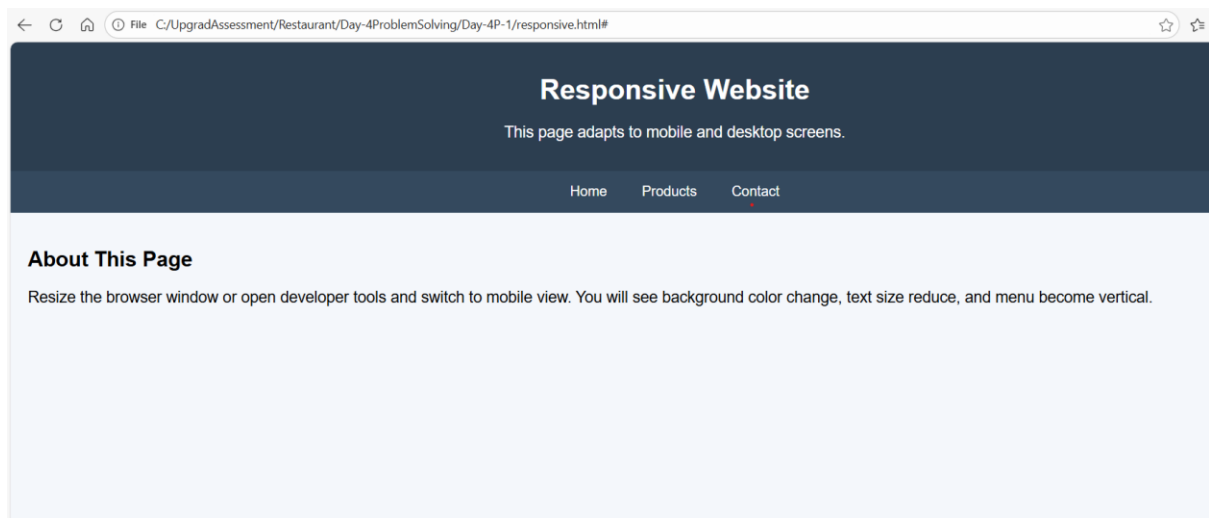
Code Snapshot:

```
File Edit View

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">

  <!-- Requirement 1: Viewport meta tag for mobile responsiveness -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Responsive Web Page - Problem 1</title>
  <style>
    /* Base styling for desktop view */
    body {
      margin: 0;
      font-family: Arial, sans-serif;
      background-color: #f4f7fb; /* Desktop background color */
    }
    header {
      background: #2c3e50;
      color: white;
      padding: 15px;
      text-align: center;
    }
    /* Desktop navigation (horizontal layout) */
    nav {
      display: flex; /* Desktop navigation in row */
      justify-content: center;
      background: #34495e;
    }
    nav a {
      color: white;
      text-decoration: none;
      padding: 15px 20px;
      display: block;
    }
    nav a:hover {
      background: #1abc9c;
    }
    main {
      padding: 20px;
    }
    h1 {
      font-size: 32px; /* Desktop font size */
    }
    p {
      font-size: 18px; /* Desktop font size */
    }
  </style>
</head>
<body>
  <header>
    <h1>Responsive Web Page</h1>
  </header>
  <nav>
    <a href="#home">Home</a>
    <a href="#about">About</a>
    <a href="#services">Services</a>
    <a href="#contact">Contact</a>
  </nav>
  <main>
    <h2>Welcome to our website</h2>
    <p>This is a responsive web page designed to work on all devices. It features a clean, modern design with a focus on user experience. The layout is flexible and adapts to different screen sizes, ensuring that the content is always easy to read and navigate. We use a combination of HTML5 and CSS3 to create a fast, reliable, and accessible web application. The design is inspired by the latest trends in web development, but with a touch of classic elegance. We hope you enjoy the look and feel of our website, and we look forward to serving you with the best possible experience. Thank you for visiting, and we hope you'll come back soon.</p>
  </main>
</body>
</html>
```

Output Snapshot:



Code Explanation:

This HTML page demonstrates responsive web design using only HTML and CSS: the `<meta name="viewport">` tag ensures the layout scales correctly on mobile devices, while the CSS defines default (desktop) styles such as background color, font sizes, and a horizontal navigation menu using Flexbox. A media query `@media (max-width: 600px)` is used to detect smaller screens (mobile), and inside it the styles change— the page background color becomes different for mobile, the heading and paragraph font sizes are reduced for better readability on small screens, and the navigation menu switches from a horizontal row to a vertical column layout. Overall, the page automatically adapts its look and layout based on screen size, making it readable and usable on both desktop and mobile devices without any JavaScript.

Problem 2:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Student Grade Evaluator</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background: #f4f7fb;

      padding: 40px;

    }

    .box {

      max-width: 400px;

      margin: auto;

      background: #ffffff;

      padding: 20px;

      border-radius: 10px;

      box-shadow: 0 4px 10px rgba(0,0,0,0.1);

    }

    input, button {

      width: 100%;

      padding: 10px;

      margin-top: 10px;

      font-size: 16px;

    }

    .result {

      margin-top: 15px;

      font-weight: bold;

    }

  </style>
```

```
</head>

<body>

  <div class="box">

    <h2>Student Grade Evaluator</h2>

    <!-- User enters marks here -->

    <input type="number" id="marksInput" placeholder="Enter student marks" />

    <button onclick="evaluateGrade()">Check Grade</button>

    <div class="result" id="result"></div>
  </div>

  <script>
    // Requirement: Use JavaScript variables (let or const)
    // Requirement: Accept the student's marks as a variable (taken from user input)
    function evaluateGrade() {
      let marks = Number(document.getElementById("marksInput").value);
      // Requirement: Use numeric data types

      let grade;
      // Requirement: Use JavaScript variable to store grade

      // Requirement: Use if-else statements
      // Requirement: Use comparison operators
      if (marks >= 75) {
        grade = "Grade A"; // Marks  $\geq$  75  $\rightarrow$  Grade A
      }
      else if (marks >= 60) {
        grade = "Grade B"; // Marks  $\geq$  60  $\rightarrow$  Grade B
      }
    }
  </script>

```

```
else if (marks >= 40) {  
    grade = "Grade C"; // Marks  $\geq$  40  $\rightarrow$  Grade C  
}  
else {  
    grade = "Fail"; // Marks < 40  $\rightarrow$  Fail  
}  
  
// Requirement: Output using document.write() or console.log()  
// Here we output on the web page (UI-friendly output)  
document.getElementById("result").innerHTML =  
    "Marks: " + marks + "<br>" +  
    "Result: " + grade;  
  
// Also logging in console for debugging  
console.log("Marks:", marks);  
console.log("Result:", grade);  
}  
</script>  
</body>  
</html>
```


Code Snapshot:

```
File Edit View

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Student Grade Evaluator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f4f7fb;
      padding: 40px;
    }
    .box {
      max-width: 400px;
      margin: auto;
      background: #ffffff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
    }
    input, button {
      width: 100%;
      padding: 10px;
      margin-top: 10px;
      font-size: 16px;
    }
    .result {
      margin-top: 15px;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div class="box">
    <h2>Student Grade Evaluator</h2>

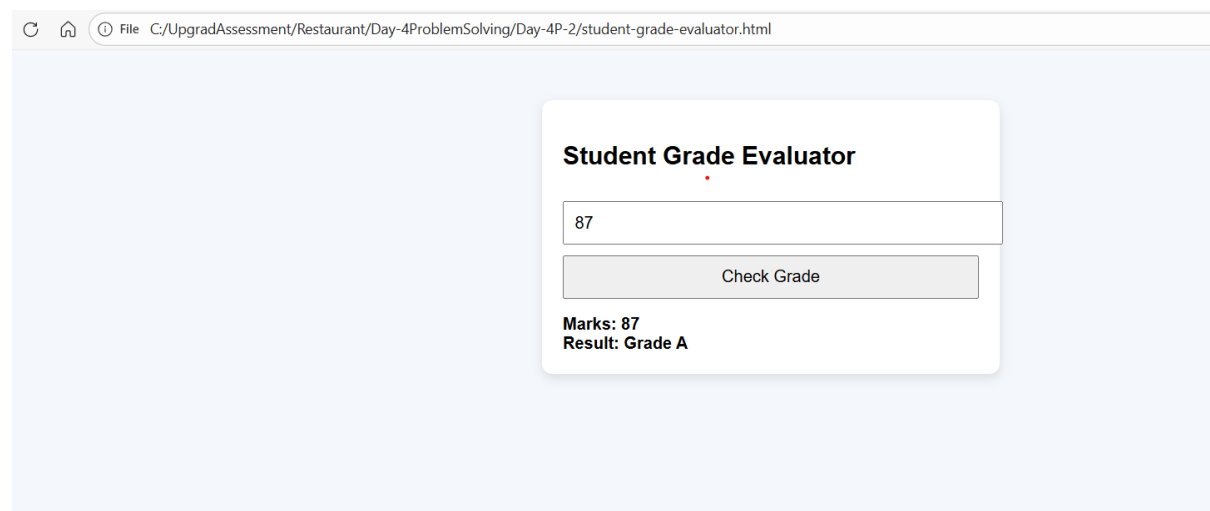
    <!-- User enters marks here -->
    <input type="number" id="marksInput" placeholder="Enter student marks" />

    <button onclick="evaluateGrade()">Check Grade</button>

    <div class="result" id="result"></div>
  </div>

  <script>
```

Output Snapshot:



Code Explanation:

This program creates a simple Student Grade Evaluator using HTML for structure, CSS for basic styling, and JavaScript for logic: the user enters marks in an input field, which are read in JavaScript using `document.getElementById()` and converted into a number using `Number()`. The marks are stored in a variable, and if-else conditions with comparison operators decide the grade (A, B, C, or Fail) based on the entered value. When the button is clicked, the `evaluateGrade()` function runs and dynamically displays the marks and calculated grade on the web page using `innerHTML`, while also logging the values to the console for debugging. This shows how variables, user input, conditions, and output work together in a simple interactive web application.

Problem 3:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Student Grade Evaluator</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background: #f4f7fb;

      padding: 40px;

    }

    .box {

      max-width: 400px;

      margin: auto;

      background: #ffffff;

      padding: 20px;

      border-radius: 10px;

      box-shadow: 0 4px 10px rgba(0,0,0,0.1);

    }

    input, button {

      width: 100%;

      padding: 10px;

      margin-top: 10px;

      font-size: 16px;

    }

    .result {

      margin-top: 15px;

      font-weight: bold;

    }

  </style>
```

```
</head>

<body>

  <div class="box">

    <h2>Student Grade Evaluator</h2>

    <!-- User enters marks here -->

    <input type="number" id="marksInput" placeholder="Enter student marks" />

    <button onclick="evaluateGrade()">Check Grade</button>

    <div class="result" id="result"></div>
  </div>

  <script>
    // Requirement: Use JavaScript variables (let or const)
    // Requirement: Accept the student's marks as a variable (taken from user input)
    function evaluateGrade() {
      let marks = Number(document.getElementById("marksInput").value);
      // Requirement: Use numeric data types

      let grade;
      // Requirement: Use JavaScript variable to store grade

      // Requirement: Use if-else statements
      // Requirement: Use comparison operators
      if (marks >= 75) {
        grade = "Grade A"; // Marks  $\geq$  75  $\rightarrow$  Grade A
      }
      else if (marks >= 60) {
        grade = "Grade B"; // Marks  $\geq$  60  $\rightarrow$  Grade B
      }
    }
  </script>

```

```
else if (marks >= 40) {  
    grade = "Grade C"; // Marks  $\geq$  40  $\rightarrow$  Grade C  
}  
else {  
    grade = "Fail"; // Marks < 40  $\rightarrow$  Fail  
}  
  
// Requirement: Output using document.write() or console.log()  
// Here we output on the web page (UI-friendly output)  
document.getElementById("result").innerHTML =  
    "Marks: " + marks + "<br>" +  
    "Result: " + grade;  
  
// Also logging in console for debugging  
console.log("Marks:", marks);  
console.log("Result:", grade);  
}  
</script>  
</body>  
</html>
```

Code Snapshot:

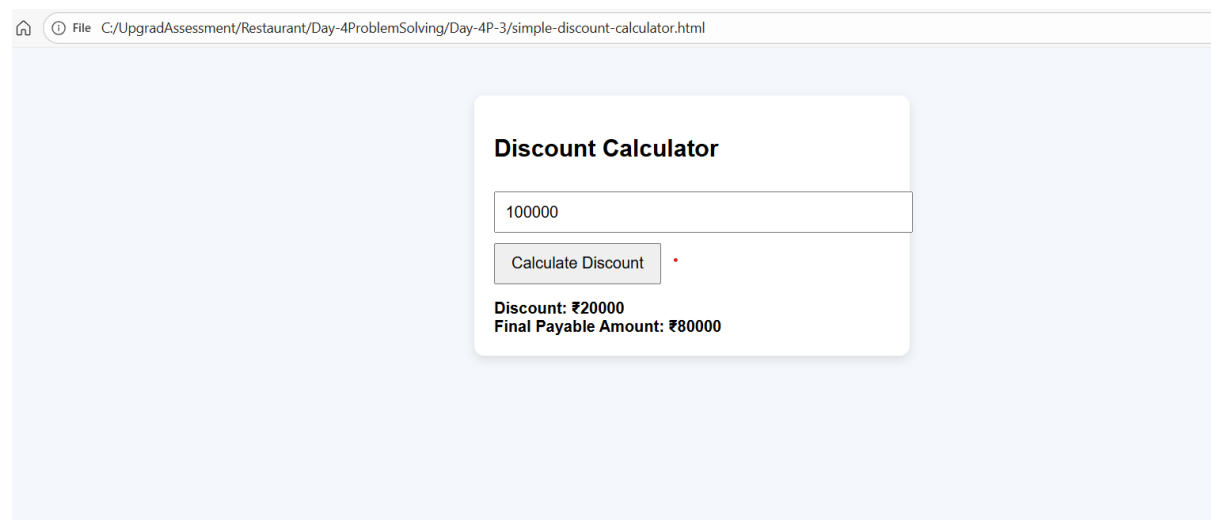
```
File Edit View

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Discount Calculator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f4f7fb;
      padding: 40px;
    }
    .box {
      max-width: 400px;
      margin: auto;
      background: #fff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
    }
    input {
      width: 100%;
      padding: 10px;
      margin: 10px 0;
      font-size: 16px;
    }
    button {
      padding: 10px 16px;
      font-size: 16px;
      cursor: pointer;
    }
    .result {
      margin-top: 15px;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div class="box">
    <h2>Discount Calculator</h2>

    <!-- User enters purchase amount here -->
    <input type="number" id="amount" placeholder="Enter purchase amount" />

    <button onclick="calculateDiscount()">Calculate Discount</button>
  </div>
</body>
</html>
```

Output Snapshot:



Code Explanation:

This program implements a simple Discount Calculator using HTML for the user interface, CSS for basic layout and styling, and JavaScript for the calculation logic: the user enters a purchase amount in an input field, which is fetched in JavaScript using `document.getElementById()` and converted into a number with `Number()`. Based on the entered amount, if-else conditions decide the discount rate (20% for ₹5000 and above, 10% for ₹3000 and above, otherwise no discount), and then the discount amount and final payable amount are calculated using arithmetic operations. When the button is clicked, the `calculateDiscount()` function runs and dynamically displays the discount and final amount on the web page using `innerHTML`, showing how variables, conditions, and calculations work together in a practical real-world example.

Problem 4:

<!--

Problem 4: Traffic Signal Simulator (Level-2)

Scenario:

A traffic control system needs a JavaScript program that displays instructions based on traffic signal color.

Requirements:

- Store signal color in a variable ("red", "yellow", "green")
- Use a switch statement to display:
 - Red → Stop
 - Yellow → Get Ready
 - Green → Go
- Handle invalid signal input gracefully

Technical Constraints:

- Must use switch–case
- Use string data types
- Use console.log() for output
- No if–else allowed

Learning Outcome:

- Use switch statements effectively
- Compare string values
- Handle multiple conditions cleanly
- Understand control flow alternatives

-->

<!DOCTYPE html>


```
<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Traffic Signal Simulator</title>

  <style>

    body {

      min-height: 100vh;

      margin: 0;

      font-family: Arial, sans-serif;

      background: linear-gradient(135deg, #1d2671, #c33764);

      display: flex;

      align-items: center;

      justify-content: center;

      color: #fff;

    }

    .card {

      background: rgba(0, 0, 0, 0.35);

      padding: 30px;

      border-radius: 16px;

      width: 380px;

      box-shadow: 0 10px 25px rgba(0,0,0,0.4);

      text-align: center;

    }

    h2 {

      margin-bottom: 8px;

      letter-spacing: 1px;

    }

    p {
```

```
font-size: 14px;  
opacity: 0.9;  
}
```

```
input {  
  width: 100%;  
  padding: 10px;  
  margin-top: 12px;  
  border-radius: 8px;  
  border: none;  
  outline: none;  
  font-size: 16px;  
}
```

```
button {  
  width: 100%;  
  margin-top: 12px;  
  padding: 10px;  
  border: none;  
  border-radius: 8px;  
  background: #ffd166;  
  color: #333;  
  font-size: 16px;  
  font-weight: bold;  
  cursor: pointer;  
}
```

```
.result {  
  margin-top: 16px;  
  padding: 10px;  
  border-radius: 8px;
```

```

    background: rgba(255, 255, 255, 0.15);
    font-size: 18px;
    font-weight: bold;
    min-height: 28px;
}
</style>
</head>
<body>
<div class="card">
    <h2>Traffic Signal Simulator</h2>
    <p>Enter signal color: red, yellow, or green</p>

    <!-- User input taken from webpage (string data type) -->
    <input type="text" id="signalInput" placeholder="e.g., red / yellow / green">

    <button onclick="showSignalInstruction()">Show Instruction</button>

    <div class="result" id="output"></div>
</div>

<script>
function showSignalInstruction() {
    // Requirement: Store signal color in a variable
    // Requirement: Use string data types
    let signalColor = document.getElementById("signalInput").value.toLowerCase();

    let message = "";

    // Requirement: Must use switch-case (No if-else allowed)
    switch (signalColor) {
        case "red":

```

```
// Requirement: Red → Stop
message = "Stop";
break;

case "yellow":
    // Requirement: Yellow → Get Ready
    message = "Get Ready";
    break;

case "green":
    // Requirement: Green → Go
    message = "Go";
    break;

default:
    // Requirement: Handle invalid signal input gracefully
    message = "Invalid signal! Please enter red, yellow, or green.";
    break;
}

// Requirement: Use console.log() for output
console.log("Signal Color:", signalColor);
console.log("Instruction:", message);

// Extra: Show result on the web page for better UI
document.getElementById("output").innerText = message;
}
</script>
</body>
</html>
```

Code Snapshot:

```
<script>
function showSignalInstruction() {
  // Requirement: Store signal color in a variable
  // Requirement: Use string data types
  let signalColor = document.getElementById("signalInput").value.toLowerCase();

  let message = "";

  // Requirement: Must use switch-case (No if-else allowed)
  switch (signalColor) {
    case "red":
      // Requirement: Red → Stop
      message = "Stop";
      break;

    case "yellow":
      // Requirement: Yellow → Get Ready
      message = "Get Ready";
      break;

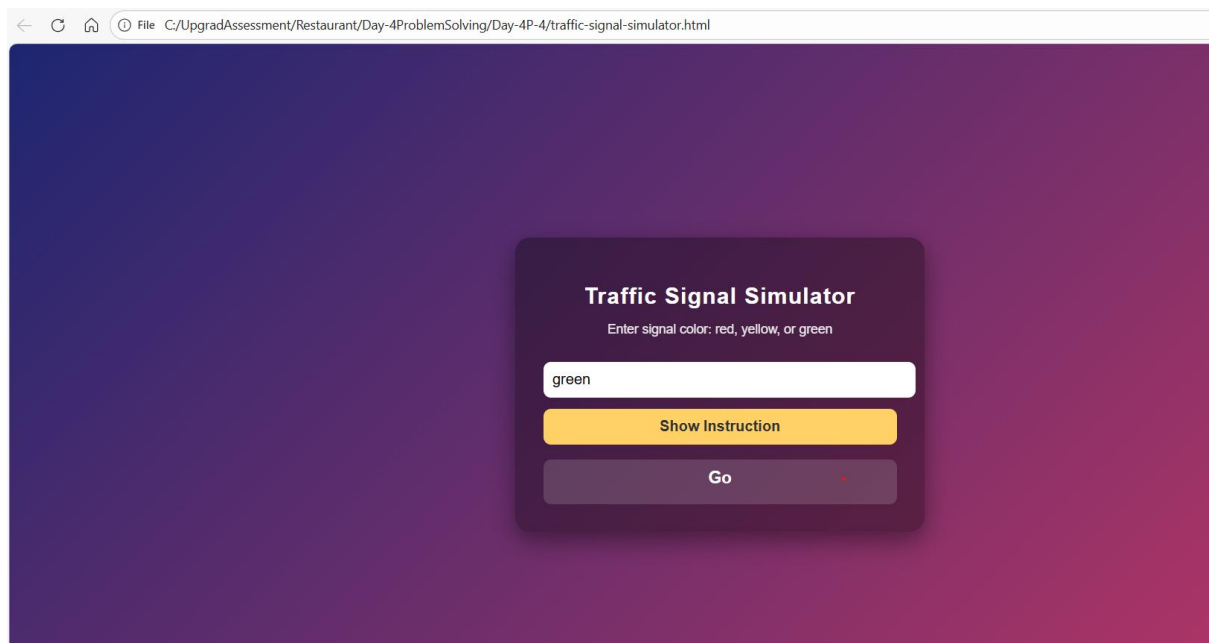
    case "green":
      // Requirement: Green → Go
      message = "Go";
      break;

    default:
      // Requirement: Handle invalid signal input gracefully
      message = "Invalid signal! Please enter red, yellow, or green.";
      break;
  }

  // Requirement: Use console.log() for output
  console.log("Signal Color:", signalColor);
  console.log("Instruction:", message);

  // Extra: Show result on the web page for better UI
  document.getElementById("output").innerText = message;
}
</script>
</body>
</html>
```

Output Snapshot:



Code Explanation:

This Traffic Signal Simulator uses HTML to create a simple input-based interface, CSS to design a modern card-style layout, and JavaScript to control the logic using a switch-case statement as required: the signal color entered by the user is stored in a string variable and normalized to lowercase, then compared against "red", "yellow", and "green" in the switch block to display the correct instruction (Stop, Get Ready, or Go), while any invalid input is handled gracefully using the default case. The result is logged to the console using `console.log()` to meet the technical constraint, and the same instruction is also shown on the web page for user-friendly feedback, demonstrating clean control flow without using any if-else conditions.

Problem 5:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <title>Number Analysis Tool</title>

  <!--
```

Problem 5: Number Analysis Tool (Level-2)

Scenario:

A utility program is required to analyze numbers and provide insights such as positivity, parity, and range.

Requirements:

- Store a number in a variable
- Use conditional (ternary) operator to check:
 - Positive or Negative
- Use if–else to check:
 - Even or Odd
- Use a loop to print all numbers from 1 to the given number

Technical Constraints:

- Store a number in a variable
- Use conditional (ternary) operator to check Positive or Negative
- Use if–else to check Even or Odd
- Use a loop to print all numbers from 1 to the given number

Learning Outcome:

- Combine multiple control flow techniques
- Use loops for iteration

- Apply conditional operators
- Build multi-step logical programs

-->

<style>

```
body {  
  min-height: 100vh;  
  margin: 0;  
  font-family: Arial, sans-serif;  
  background: linear-gradient(135deg, #232526, #414345);  
  display: flex;  
  align-items: center;  
  justify-content: center;  
  color: #fff;  
}
```

```
.card {  
  background: rgba(0, 0, 0, 0.45);  
  padding: 28px;  
  border-radius: 16px;  
  width: 420px;  
  box-shadow: 0 10px 25px rgba(0,0,0,0.4);  
}
```

```
h2 {  
  text-align: center;  
  margin-bottom: 8px;  
  letter-spacing: 1px;  
}
```

```
p {
```



```
text-align: center;

font-size: 14px;

opacity: 0.9;
}
```

```
input {

width: 100%;

padding: 10px;

margin-top: 12px;

border-radius: 8px;

border: none;

outline: none;

font-size: 16px;
}
```

```
button {

width: 100%;

margin-top: 12px;

padding: 10px;

border: none;

border-radius: 8px;

background: #00e5ff;

color: #00323a;

font-size: 16px;

font-weight: bold;

cursor: pointer;
}
```

```
.result {

margin-top: 14px;

padding: 10px;
```

```
border-radius: 8px;
background: rgba(255, 255, 255, 0.12);
min-height: 24px;
font-weight: bold;
}
```

```
.list {
margin-top: 10px;
max-height: 120px;
overflow-y: auto;
background: rgba(0,0,0,0.3);
padding: 8px;
border-radius: 8px;
font-size: 14px;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div class="card">
```

```
<h2>Number Analysis Tool</h2>
```

```
<p>Enter any number to analyze</p>
```

```
<!-- User input from webpage -->
```

```
<input type="number" id="numInput" placeholder="Enter a number">
```

```
<button onclick="analyzeNumber()">Analyze Number</button>
```

```
<div class="result" id="signResult"></div>
```

```
<div class="result" id="parityResult"></div>
```

```
<div class="list" id="rangeResult"></div>
```

```
</div>
```

```
<script>
```

```
function analyzeNumber() {
```

```
    // Requirement: Store a number in a variable
```

```
    let number = Number(document.getElementById("numInput").value);
```

```
    // Requirement: Use conditional (ternary) operator to check Positive or Negative
```

```
    let signResult = (number >= 0) ? "Positive Number" : "Negative Number";
```

```
    // Requirement: Use if-else to check Even or Odd
```

```
    let parityResult = "";
```

```
    if (number % 2 === 0) {
```

```
        parityResult = "Even Number";
```

```
    } else {
```

```
        parityResult = "Odd Number";
```

```
    }
```

```
    // Requirement: Use a loop to print all numbers from 1 to the given number
```

```
    let output = "";
```

```
    for (let i = 1; i <= number; i++) {
```

```
        output += i + " ";
```

```
    }
```

```
    // Display results on webpage
```

```
    document.getElementById("signResult").innerText = "Sign: " + signResult;
```

```
    document.getElementById("parityResult").innerText = "Parity: " + parityResult;
```

```
    document.getElementById("rangeResult").innerText = "Numbers from 1 to " + number + ": " +  
output;
```

```
    // Console output (for debugging / learning)
```

```
    console.log("Number:", number);
```

```

        console.log("Sign:", signResult);

        console.log("Parity:", parityResult);

        console.log("Range:", output);

    }

</script>

</body>

</html>

```

Code Snapshot:

```

File    Edit    View

<button onclick="analyzeNumber()">Analyze Number</button>

<div class="result" id="signResult"></div>
<div class="result" id="parityResult"></div>
<div class="list" id="rangeResult"></div>
</div>

<script>
function analyzeNumber() {
    // Requirement: Store a number in a variable
    let number = Number(document.getElementById("numInput").value);

    // Requirement: Use conditional (ternary) operator to check Positive or Negative
    let signResult = (number >= 0) ? "Positive Number" : "Negative Number";

    // Requirement: Use if-else to check Even or Odd
    let parityResult = "";
    if (number % 2 === 0) {
        parityResult = "Even Number";
    } else {
        parityResult = "Odd Number";
    }

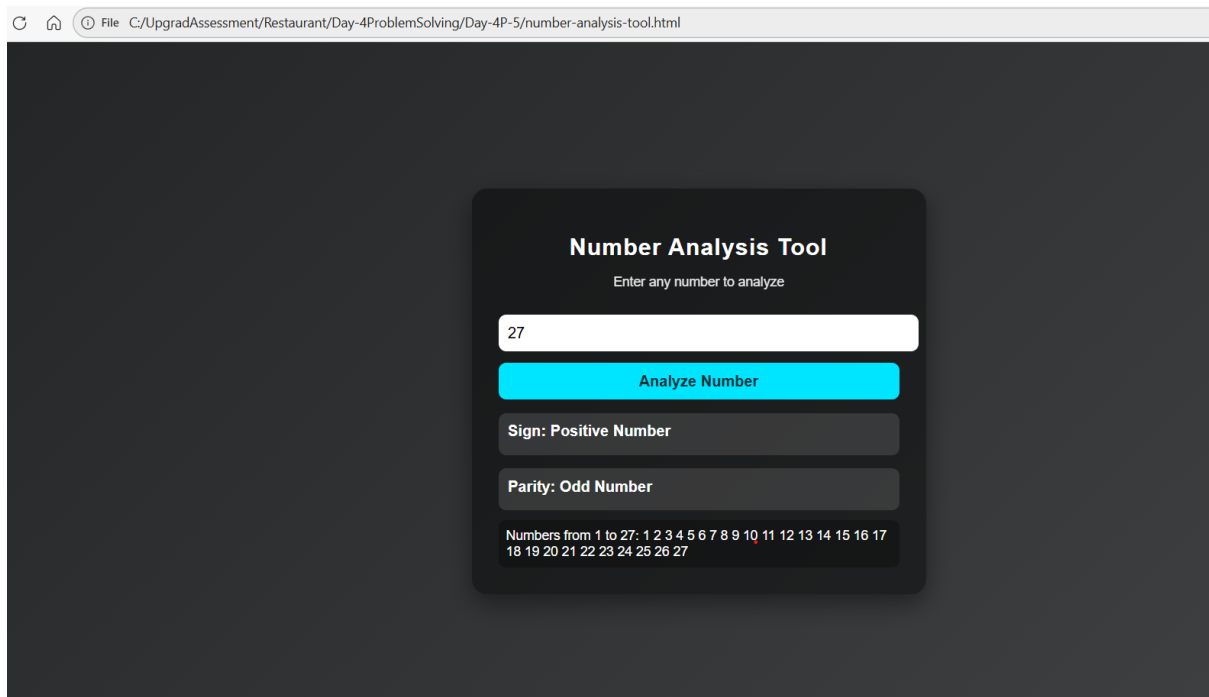
    // Requirement: Use a loop to print all numbers from 1 to the given number
    let output = "";
    for (let i = 1; i <= number; i++) {
        output += i + " ";
    }

    // Display results on webpage
    document.getElementById("signResult").innerText = "Sign: " + signResult;
    document.getElementById("parityResult").innerText = "Parity: " + parityResult;
    document.getElementById("rangeResult").innerText = "Numbers from 1 to " + number + ": " + output;

    // Console output (for debugging / learning)
    console.log("Number:", number);
    console.log("Sign:", signResult);
    console.log("Parity:", parityResult);
    console.log("Range:", output);
}
</script>
</body>
</html>

```

Output Snapshot:



Code Explanation:

This Number Analysis Tool combines multiple JavaScript control flow concepts in one small program: the user enters a value which is stored in a variable after converting it to a number, then a ternary (conditional) operator checks whether the number is positive or negative, an if-else statement determines whether the number is even or odd using the modulus operator, and finally a for loop iterates from 1 up to the given number to generate a sequence of numbers. The results of each step are displayed dynamically on the web page for easy understanding and also logged to the console for learning and debugging, showing how different decision-making and looping techniques work together in a single, real-world utility program.