Double-click (or enter) to edit

```
Start coding or generate with AI.
```

1. In the following shopping cart add, remove, and edit items

   => const shoppingCart = ['Milk', 'Coffee', 'Tea', 'Honey'O L add 'Meat' in the beginning of your shopping cart if it has not been already addeJ L add Sugar at the end of you shopping cart if it has not been already addeJ L remove 'Honey' if you are allergic to hone L modify Tea to 'Green Tea'

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Analytics Dashboard | EduMetrics Pro</title>
    <style>
        :root {
            --primary-color: #1a365d;
            --secondary-color: #2c5282;
            --accent-color: #3182ce;
            --success-color: #38a169;
            --warning-color: #d69e2e;
            --error-color: #e53e3e;
            --neutral-100: #f7fafc;
            --neutral-200: #edf2f7;
            --neutral-300: #e2e8f0;
            --neutral-400: #cbd5e0;
            --neutral-500: #a0aec0;
            --neutral-600: #718096;
            --neutral-700: #4a5568;
            --neutral-800: #2d3748;
            --neutral-900: #1a202c;
            --shadow-sm: 0 1px 2px 0 rgba(0, 0, 0, 0.05);
            --shadow-md: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.06);
            --shadow-lg: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);
            --shadow-xl: 0 20px 25px -5px rgba(0, 0, 0, 0.1), 0 10px 10px -5px rgba(0, 0, 0, 0.04);
        }

        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Inter', -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, sans-serif;
            background: linear-gradient(135deg, var(--neutral-100) 0%, var(--neutral-200) 100%);
            min-height: 100vh;
            color: var(--neutral-800);
            line-height: 1.6;
        }
```

```
.header {
    background: linear-gradient(135deg, var(--primary-color) 0%, var(--secondary-color) 100%);
    color: white;
    padding: 2rem 0;
    box-shadow: var(--shadow-lg);
    position: relative;
    overflow: hidden;
}

.header::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100"><defs><pattern id="grid" width="10" height="10" patternUnits="userSpa
    opacity: 0.3;
}

.header-content {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 2rem;
    text-align: center;
    position: relative;
    z-index: 1;
}

.header h1 {
    font-size: 2.5rem;
    font-weight: 700;
    margin-bottom: 0.5rem;
    letter-spacing: -0.025em;
}

.header p {
    font-size: 1.2rem;
    opacity: 0.9;
    font-weight: 300;
}

.main-container {
    max-width: 1200px;
    margin: -2rem auto 0;
    padding: 0 2rem 4rem;
    position: relative;
    z-index: 2;
}

.card {
    background: white;
    border-radius: 16px;
    box-shadow: var(--shadow-xl);
    border: 1px solid var(--neutral-200);
    overflow: hidden;
    transition: all 0.3s ease;
}

.card:hover {
    box-shadow: var(--shadow-xl), 0 0 0 1px var(--accent-color);
}

.card-header {
    background: linear-gradient(135deg, var(--neutral-50) 0%, var(--neutral-100) 100%);
    padding: 2rem;
    border-bottom: 1px solid var(--neutral-200);
}

.card-header h2 {
    color: var(--neutral-800);
    font-size: 1.5rem;
    font-weight: 600;
    margin-bottom: 0.5rem;
    display: flex;
    align-items: center;
    gap: 0.75rem;
}

.card-header p {
    color: var(--neutral-600);
    font-size: 0.95rem;
}

.card-body {
    padding: 2rem;
}

.form-group {
    margin-bottom: 2rem;
}

.form-label {
    display: block;
    font-weight: 600;
    color: var(--neutral-700);
    margin-bottom: 0.5rem;
    font-size: 0.95rem;
}

.form-input {
    width: 100%;
    padding: 1rem 1.25rem;
    border: 2px solid var(--neutral-300);
    border-radius: 8px;
    font-size: 1rem;
    background: white;
    transition: all 0.2s ease;
    font-family: 'Inter', sans-serif;
}
```

```
.form-input:focus {
    outline: none;
    border-color: var(--accent-color);
    box-shadow: 0 0 0 3px rgba(49, 130, 206, 0.1);
}

.form-input:hover {
    border-color: var(--neutral-400);
}

.button-group {
    display: flex;
    gap: 1rem;
    margin-bottom: 2rem;
}

.btn {
    flex: 1;
    padding: 1rem 2rem;
    border: none;
    border-radius: 8px;
    font-size: 1rem;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.2s ease;
    text-transform: uppercase;
    letter-spacing: 0.025em;
    font-family: 'Inter', sans-serif;
    position: relative;
    overflow: hidden;
}

.btn::before {
    content: '';
    position: absolute;
    top: 50%;
    left: 50%;
    width: 0;
    height: 0;
    background: rgba(255, 255, 255, 0.2);
    border-radius: 50%;
    transform: translate(-50%, -50%);
    transition: all 0.3s ease;
}

.btn:hover::before {
    width: 300px;
    height: 300px;
}

.btn-primary {
    background: linear-gradient(135deg, var(--accent-color) 0%, var(--secondary-color) 100%);
    color: white;
    box-shadow: var(--shadow-md);
}
```

```
.btn-primary:hover {
    transform: translateY(-2px);
    box-shadow: var(--shadow-lg);
}

.btn-secondary {
    background: linear-gradient(135deg, var(--neutral-600) 0%, var(--neutral-700) 100%);
    color: white;
    box-shadow: var(--shadow-md);
}

.btn-secondary:hover {
    transform: translateY(-2px);
    box-shadow: var(--shadow-lg);
}

.results-container {
    margin-top: 2rem;
    animation: slideUp 0.5s ease-out;
}

@keyframes slideUp {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));
    gap: 1.5rem;
    margin-bottom: 2rem;
}

.stat-card {
    background: white;
    border: 1px solid var(--neutral-200);
    border-radius: 12px;
    padding: 1.5rem;
    box-shadow: var(--shadow-sm);
    transition: all 0.2s ease;
}

.stat-card:hover {
    box-shadow: var(--shadow-md);
    transform: translateY(-2px);
}

.stat-header {
    display: flex;
    align-items: center;
    justify-content: space-between;
    margin-bottom: 1rem;
}

.stat-title {
    font-size: 0.9rem;
    font-weight: 600;
    color: var(--neutral-600);
    text-transform: uppercase;
    letter-spacing: 0.05em;
}

.stat-icon {
    width: 24px;
    height: 24px;
    background: var(--accent-color);
    border-radius: 6px;
    display: flex;
    align-items: center;
    justify-content: center;
    color: white;
    font-size: 0.8rem;
    font-weight: 600;
}

.stat-value {
    font-size: 2rem;
    font-weight: 700;
    color: var(--neutral-800);
    margin-bottom: 0.5rem;
}

.stat-description {
    font-size: 0.85rem;
    color: var(--neutral-600);
    line-height: 1.4;
}

.comparison-card {
    background: linear-gradient(135deg, var(--primary-color) 0%, var(--secondary-color) 100%);
    color: white;
    border-radius: 12px;
    padding: 2rem;
    margin-top: 1.5rem;
    position: relative;
    overflow: hidden;
}

.comparison-card::before {
    content: '';
    position: absolute;
    top: 0;
    right: 0;
    width: 100px;
    height: 100px;
    background: rgba(255, 255, 255, 0.1);
```

```
            border-radius: 50%;
            transform: translate(30px, -30px);
        }

        .comparison-title {
            font-size: 1.2rem;
            font-weight: 600;
            margin-bottom: 1rem;
            position: relative;
        }

        .comparison-content {
            display: grid;
            grid-template-columns: 1fr 1fr;
            gap: 2rem;
            margin-bottom: 1.5rem;
        }

        .comparison-item {
            text-align: center;
        }

        .comparison-label {
            font-size: 0.9rem;
            opacity: 0.8;
            margin-bottom: 0.5rem;
        }

        .comparison-value {
            font-size: 1.8rem;
            font-weight: 700;
            margin-bottom: 0.25rem;
        }

        .comparison-result {
            background: rgba(255, 255, 255, 0.1);
            border-radius: 8px;
            padding: 1rem;
            text-align: center;
            font-weight: 600;
            position: relative;
        }

        .error-message {
            background: linear-gradient(135deg, var(--error-color) 0%, #c53030 100%);
            color: white;
            padding: 1rem 1.5rem;
            border-radius: 8px;
            margin-top: 1rem;
            font-weight: 500;
        }

        .data-table {
            width: 100%;
            border-collapse: collapse;
            margin-top: 1rem;
```

```
}

.data-table th,
.data-table td {
    padding: 0.75rem 1rem;
    text-align: left;
    border-bottom: 1px solid var(--neutral-200);
}

.data-table th {
    background: var(--neutral-50);
    font-weight: 600;
    color: var(--neutral-700);
    font-size: 0.9rem;
    text-transform: uppercase;
    letter-spacing: 0.05em;
}

.data-table td {
    font-family: 'Monaco', 'Menlo', monospace;
    font-size: 0.9rem;
}

@media (max-width: 768px) {
    .header h1 {
        font-size: 2rem;
    }

    .header p {
        font-size: 1rem;
    }

    .main-container {
        padding: 0 1rem 2rem;
    }

    .card-header,
    .card-body {
        padding: 1.5rem;
    }

    .button-group {
        flex-direction: column;
    }

    .stats-grid {
        grid-template-columns: 1fr;
    }

    .comparison-content {
        grid-template-columns: 1fr;
        gap: 1rem;
    }
}

@media (max-width: 480px) {
```

```
            .header-content {
                padding: 0 1rem;
            }

            .stat-value {
                font-size: 1.5rem;
            }

            .comparison-value {
                font-size: 1.4rem;
            }
        }
    </style>
</head>
<body>
    <header class="header">
        <div class="header-content">
            <h1>EduMetrics Pro</h1>
            <p>Advanced Student Analytics Dashboard</p>
        </div>
    </header>

    <main class="main-container">
        <div class="card">
            <div class="card-header">
                <h2>
                    <span class="stat-icon">📊</span>
                    Statistical Analysis Tool
                </h2>
                <p>Comprehensive statistical analysis of student age demographics with professional-grade calculations</p>
            </div>

            <div class="card-body">
                <div class="form-group">
                    <label for="ages-input" class="form-label">Student Ages Dataset</label>
                    <input
                        type="text"
                        id="ages-input"
                        class="form-input"
                        placeholder="Enter comma-separated age values (e.g., 19, 22, 19, 24, 20, 25, 26, 24, 25, 24)"
                        value="19, 22, 19, 24, 20, 25, 26, 24, 25, 24"
                    >
                </div>

                <div class="button-group">
                    <button class="btn btn-primary" onclick="calculateStats()">
                        Analyze Dataset
                    </button>
                    <button class="btn btn-secondary" onclick="resetAnalysis()">
                        Clear Analysis
                    </button>
                </div>

                <div id="results" class="results-container" style="display: none;">
                    <div class="stats-grid" id="stats-grid">
                        <!-- Stats will be populated here -->
                    </div>

                    <div class="comparison-card">
                        <h3 class="comparison-title">📈 Variance Analysis</h3>
                        <div class="comparison-content" id="comparison-content">
                            <!-- Comparison will be populated here -->
                        </div>
                        <div class="comparison-result" id="comparison-result">
                            <!-- Result will be populated here -->
                        </div>
                    </div>

                    <div class="card" style="margin-top: 2rem;">
                        <div class="card-header">
                            <h2>
                                <span class="stat-icon">📋</span>
                                Data Summary
                            </h2>
                        </div>
                        <div class="card-body">
                            <table class="data-table">
                                <thead>
                                    <tr>
                                        <th>Dataset</th>
                                        <th>Values</th>
                                    </tr>
                                </thead>
                                <tbody id="data-table-body">
                                    <!-- Table data will be populated here -->
                                </tbody>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </main>

    <script>
        function calculateStats() {
            const input = document.getElementById('ages-input').value.trim();

            if (!input) {
                showError('Please enter student ages to analyze.');
                return;
            }

            try {
                // Parse and validate input
                const ages = input.split(',').map(age => {
                    const num = parseInt(age.trim());
                    if (isNaN(num) || num <= 0 || num > 120) {
                        throw new Error('Invalid age value');
                    }
                    return num;
                });
                if (ages.length === 0) {
                    throw new Error('No valid ages found');
                }

                // Perform calculations
                const sortedAges = [...ages].sort((a, b) => a - b);
                const minAge = Math.min(...ages);
                const maxAge = Math.max(...ages);
                const median = calculateMedian(sortedAges);
                const average = ages.reduce((sum, age) => sum + age, 0) / ages.length;
                const range = maxAge - minAge;
                const minDistance = Math.abs(minAge - average);
                const maxDistance = Math.abs(maxAge - average);

                // Display results
                displayResults({
                    original: ages,
                    sorted: sortedAges,
                    min: minAge,
                    max: maxAge,
                    median: median,
                    average: average,
                    range: range,
                    minDistance: minDistance,
                    maxDistance: maxDistance,
                    count: ages.length
                });

            } catch (error) {
                showError('Invalid input format. Please ensure all values are valid ages between 1-120, separated by commas.');
            }
        }

        function calculateMedian(sortedArray) {
            const length = sortedArray.length;
            const middle = Math.floor(length / 2);

            if (length % 2 === 0) {
                return (sortedArray[middle - 1] + sortedArray[middle]) / 2;
            } else {
                return sortedArray[middle];
            }
        }

        function displayResults(stats) {
            const resultsDiv = document.getElementById('results');
            const statsGrid = document.getElementById('stats-grid');
            const comparisonContent = document.getElementById('comparison-content');
            const comparisonResult = document.getElementById('comparison-result');
            const dataTableBody = document.getElementById('data-table-body');

            // Populate stats grid
            statsGrid.innerHTML = `
                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Sample Size</div>
                        <div class="stat-icon">N</div>
                    </div>
                    <div class="stat-value">${stats.count}</div>
                    <div class="stat-description">Total number of students in dataset</div>
                </div>

                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Minimum Age</div>
                        <div class="stat-icon">↓</div>
                    </div>
                    <div class="stat-value">${stats.min}</div>
                    <div class="stat-description">Youngest student in the dataset</div>
                </div>

                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Maximum Age</div>
                        <div class="stat-icon">↑</div>
                    </div>
                    <div class="stat-value">${stats.max}</div>
                    <div class="stat-description">Oldest student in the dataset</div>
                </div>

                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Median Age</div>
                        <div class="stat-icon">⌀</div>
                    </div>
                    <div class="stat-value">${stats.median}</div>
                    <div class="stat-description">Middle value when sorted</div>
                </div>

                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Average Age</div>
                        <div class="stat-icon">μ</div>
                    </div>
                    <div class="stat-value">${stats.average.toFixed(2)}</div>
                    <div class="stat-description">Arithmetic mean of all ages</div>
                </div>

                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Range</div>
                        <div class="stat-icon">R</div>
                    </div>
                    <div class="stat-value">${stats.range}</div>
                    <div class="stat-description">Difference between max and min</div>
                </div>
            `;

            // Populate comparison
            comparisonContent.innerHTML = `
                <div class="comparison-item">
```

```
                    <div class="comparison-label">|min - average|</div>
                    <div class="comparison-value">${stats.minDistance.toFixed(2)}</div>
                </div>
                <div class="comparison-item">
                    <div class="comparison-label">|max - average|</div>
                    <div class="comparison-value">${stats.maxDistance.toFixed(2)}</div>
                </div>
            `;

            const comparisonText = stats.minDistance < stats.maxDistance
                ? 'Minimum age is closer to the average than maximum age'
                : stats.minDistance > stats.maxDistance
                ? 'Maximum age is closer to the average than minimum age'
                : 'Both minimum and maximum ages are equally distant from the average';

            comparisonResult.innerHTML = `
                <strong>Analysis Result:</strong><br>
                ${comparisonText}
            `;

            // Populate data table
            dataTableBody.innerHTML = `
                <tr>
                    <td><strong>Original Dataset</strong></td>
                    <td>[${stats.original.join(', ')}]</td>
                </tr>
                <tr>
                    <td><strong>Sorted Dataset</strong></td>
                    <td>[${stats.sorted.join(', ')}]</td>
                </tr>
            `;

            resultsDiv.style.display = 'block';
        }

        function showError(message) {
            const resultsDiv = document.getElementById('results');
            const statsGrid = document.getElementById('stats-grid');

            statsGrid.innerHTML = `<div class="error-message">${message}</div>`;
            resultsDiv.style.display = 'block';
        }

        function resetAnalysis() {
            document.getElementById('ages-input').value = '';
            document.getElementById('results').style.display = 'none';
        }

        // Auto-calculate on page load
        window.addEventListener('load', function() {
            calculateStats();
        });
    </script>
</body>
</html>
```

The following is an array of 10 students ages: => const ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24O L Sort the array and find the min and max age1 L Find the median age(one middle item or two middle items divided by two m L Find the average age(all items divided by number of items m L Find the range of the ages(max minus min m L Compare the value of (min - average) and (max - average), use abs() method

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Analytics Dashboard | EduMetrics Pro</title>
    <style>
        :root {
            --primary-color: #1a365d;
            --secondary-color: #2c5282;
            --accent-color: #3182ce;
            --success-color: #38a169;
            --warning-color: #d69e2e;
            --error-color: #e53e3e;
            --neutral-100: #f7fafc;
            --neutral-200: #edf2f7;
            --neutral-300: #e2e8f0;
            --neutral-400: #cbd5e0;
            --neutral-500: #a0aec0;
            --neutral-600: #718096;
            --neutral-700: #4a5568;
            --neutral-800: #2d3748;
            --neutral-900: #1a202c;
            --shadow-sm: 0 1px 2px 0 rgba(0, 0, 0, 0.05);
            --shadow-md: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.06);
            --shadow-lg: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);
            --shadow-xl: 0 20px 25px -5px rgba(0, 0, 0, 0.1), 0 10px 10px -5px rgba(0, 0, 0, 0.04);
        }

        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Inter', -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, sans-serif;
            background: linear-gradient(135deg, var(--neutral-100) 0%, var(--neutral-200) 100%);
            min-height: 100vh;
            color: var(--neutral-800);
            line-height: 1.6;
        }

        .header {
            background: linear-gradient(135deg, var(--primary-color) 0%, var(--secondary-color) 100%);
            color: white;
            padding: 2rem 0;
            box-shadow: var(--shadow-lg);
            position: relative;
    overflow: hidden;
}

.header::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    background: url('data:image/svg+xml,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100"><defs><pattern id="grid" width="10" height="10" patternUnits="userSpa
    opacity: 0.3;
}

.header-content {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 2rem;
    text-align: center;
    position: relative;
    z-index: 1;
}

.header h1 {
    font-size: 2.5rem;
    font-weight: 700;
    margin-bottom: 0.5rem;
    letter-spacing: -0.025em;
}

.header p {
    font-size: 1.2rem;
    opacity: 0.9;
    font-weight: 300;
}

.main-container {
    max-width: 1200px;
    margin: -2rem auto 0;
    padding: 0 2rem 4rem;
    position: relative;
    z-index: 2;
}

.card {
    background: white;
    border-radius: 16px;
    box-shadow: var(--shadow-xl);
    border: 1px solid var(--neutral-200);
    overflow: hidden;
    transition: all 0.3s ease;
}

.card:hover {
    box-shadow: var(--shadow-xl), 0 0 0 1px var(--accent-color);
}
```

```css
.card-header {
    background: linear-gradient(135deg, var(--neutral-50) 0%, var(--neutral-100) 100%);
    padding: 2rem;
    border-bottom: 1px solid var(--neutral-200);
}

.card-header h2 {
    color: var(--neutral-800);
    font-size: 1.5rem;
    font-weight: 600;
    margin-bottom: 0.5rem;
    display: flex;
    align-items: center;
    gap: 0.75rem;
}

.card-header p {
    color: var(--neutral-600);
    font-size: 0.95rem;
}

.card-body {
    padding: 2rem;
}

.form-group {
    margin-bottom: 2rem;
}

.form-label {
    display: block;
    font-weight: 600;
    color: var(--neutral-700);
    margin-bottom: 0.5rem;
    font-size: 0.95rem;
}

.form-input {
    width: 100%;
    padding: 1rem 1.25rem;
    border: 2px solid var(--neutral-300);
    border-radius: 8px;
    font-size: 1rem;
    background: white;
    transition: all 0.2s ease;
    font-family: 'Inter', sans-serif;
}

.form-input:focus {
    outline: none;
    border-color: var(--accent-color);
    box-shadow: 0 0 0 3px rgba(49, 130, 206, 0.1);
}

.form-input:hover {
    border-color: var(--neutral-400);
}

.button-group {
    display: flex;
    gap: 1rem;
    margin-bottom: 2rem;
}

.btn {
    flex: 1;
    padding: 1rem 2rem;
    border: none;
    border-radius: 8px;
    font-size: 1rem;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.2s ease;
    text-transform: uppercase;
    letter-spacing: 0.025em;
    font-family: 'Inter', sans-serif;
    position: relative;
    overflow: hidden;
}

.btn::before {
    content: '';
    position: absolute;
    top: 50%;
    left: 50%;
    width: 0;
    height: 0;
    background: rgba(255, 255, 255, 0.2);
    border-radius: 50%;
    transform: translate(-50%, -50%);
    transition: all 0.3s ease;
}

.btn:hover::before {
    width: 300px;
    height: 300px;
}

.btn-primary {
    background: linear-gradient(135deg, var(--accent-color) 0%, var(--secondary-color) 100%);
    color: white;
    box-shadow: var(--shadow-md);
}

.btn-primary:hover {
    transform: translateY(-2px);
    box-shadow: var(--shadow-lg);
}

.btn-secondary {
    background: linear-gradient(135deg, var(--neutral-600) 0%, var(--neutral-700) 100%);
    color: white;
    box-shadow: var(--shadow-md);
}

.btn-secondary:hover {
    transform: translateY(-2px);
    box-shadow: var(--shadow-lg);
}

.results-container {
    margin-top: 2rem;
    animation: slideUp 0.5s ease-out;
}

@keyframes slideUp {
    from {
        opacity: 0;
        transform: translateY(20px);
    }
    to {
        opacity: 1;
        transform: translateY(0);
    }
}

.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));
    gap: 1.5rem;
    margin-bottom: 2rem;
}

.stat-card {
    background: white;
    border: 1px solid var(--neutral-200);
    border-radius: 12px;
    padding: 1.5rem;
    box-shadow: var(--shadow-sm);
    transition: all 0.2s ease;
}

.stat-card:hover {
    box-shadow: var(--shadow-md);
    transform: translateY(-2px);
}

.stat-header {
    display: flex;
    align-items: center;
    justify-content: space-between;
    margin-bottom: 1rem;
}

.stat-title {
    font-size: 0.9rem;
    font-weight: 600;
    color: var(--neutral-600);
    text-transform: uppercase;
    letter-spacing: 0.05em;
}

.stat-icon {
    width: 24px;
    height: 24px;
    background: var(--accent-color);
    border-radius: 6px;
    display: flex;
    align-items: center;
    justify-content: center;
    color: white;
    font-size: 0.8rem;
    font-weight: 600;
}

.stat-value {
    font-size: 2rem;
    font-weight: 700;
    color: var(--neutral-800);
    margin-bottom: 0.5rem;
}

.stat-description {
    font-size: 0.85rem;
    color: var(--neutral-600);
    line-height: 1.4;
}

.comparison-card {
    background: linear-gradient(135deg, var(--primary-color) 0%, var(--secondary-color) 100%);
    color: white;
    border-radius: 12px;
    padding: 2rem;
    margin-top: 1.5rem;
    position: relative;
    overflow: hidden;
}

.comparison-card::before {
    content: '';
    position: absolute;
    top: 0;
    right: 0;
    width: 100px;
    height: 100px;
    background: rgba(255, 255, 255, 0.1);
    border-radius: 50%;
    transform: translate(30px, -30px);
}

.comparison-title {
    font-size: 1.2rem;
    font-weight: 600;
    margin-bottom: 1rem;
    position: relative;
}

.comparison-content {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 2rem;
    margin-bottom: 1.5rem;
}

.comparison-item {
    text-align: center;
}

.comparison-label {
    font-size: 0.9rem;
    opacity: 0.8;
    margin-bottom: 0.5rem;
}

.comparison-value {
    font-size: 1.8rem;
    font-weight: 700;
    margin-bottom: 0.25rem;
}

.comparison-result {
    background: rgba(255, 255, 255, 0.1);
    border-radius: 8px;
    padding: 1rem;
    text-align: center;
    font-weight: 600;
    position: relative;
}

.error-message {
    background: linear-gradient(135deg, var(--error-color) 0%, #c53030 100%);
    color: white;
    padding: 1rem 1.5rem;
    border-radius: 8px;
    margin-top: 1rem;
    font-weight: 500;
}

.data-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 1rem;
}

.data-table th,
.data-table td {
    padding: 0.75rem 1rem;
    text-align: left;
    border-bottom: 1px solid var(--neutral-200);
```

```
}

.data-table th {
    background: var(--neutral-50);
    font-weight: 600;
    color: var(--neutral-700);
    font-size: 0.9rem;
    text-transform: uppercase;
    letter-spacing: 0.05em;
}

.data-table td {
    font-family: 'Monaco', 'Menlo', monospace;
    font-size: 0.9rem;
}

@media (max-width: 768px) {
    .header h1 {
        font-size: 2rem;
    }

    .header p {
        font-size: 1rem;
    }

    .main-container {
        padding: 0 1rem 2rem;
    }

    .card-header,
    .card-body {
        padding: 1.5rem;
    }

    .button-group {
        flex-direction: column;
    }

    .stats-grid {
        grid-template-columns: 1fr;
    }

    .comparison-content {
        grid-template-columns: 1fr;
        gap: 1rem;
    }
}

@media (max-width: 480px) {
    .header-content {
        padding: 0 1rem;
    }

    .stat-value {
        font-size: 1.5rem;
    }
```

```
            .comparison-value {
                font-size: 1.4rem;
            }
        }
    </style>
</head>
<body>
    <header class="header">
        <div class="header-content">
            <h1>EduMetrics Pro</h1>
            <p>Advanced Student Analytics Dashboard</p>
        </div>
    </header>

    <main class="main-container">
        <div class="card">
            <div class="card-header">
                <h2>
                    <span class="stat-icon">📊</span>
                    Statistical Analysis Tool
                </h2>
                <p>Comprehensive statistical analysis of student age demographics with professional-grade calculations</p>
            </div>

            <div class="card-body">
                <div class="form-group">
                    <label for="ages-input" class="form-label">Student Ages Dataset</label>
                    <input
                        type="text"
                        id="ages-input"
                        class="form-input"
                        placeholder="Enter comma-separated age values (e.g., 19, 22, 19, 24, 20, 25, 26, 24, 25, 24)"
                        value="19, 22, 19, 24, 20, 25, 26, 24, 25, 24"
                    >
                </div>

                <div class="button-group">
                    <button class="btn btn-primary" onclick="calculateStats()">
                        Analyze Dataset
                    </button>
                    <button class="btn btn-secondary" onclick="resetAnalysis()">
                        Clear Analysis
                    </button>
                </div>

                <div id="results" class="results-container" style="display: none;">
                    <div class="stats-grid" id="stats-grid">
                        <!-- Stats will be populated here -->
                    </div>

                    <div class="comparison-card">
                        <h3 class="comparison-title">📈 Variance Analysis</h3>
                        <div class="comparison-content" id="comparison-content">
                            <!-- Comparison will be populated here -->
                        </div>
                        <div class="comparison-result" id="comparison-result">
                            <!-- Result will be populated here -->
                        </div>
                    </div>

                    <div class="card" style="margin-top: 2rem;">
                        <div class="card-header">
                            <h2>
                                <span class="stat-icon">📋</span>
                                Data Summary
                            </h2>
                        </div>
                        <div class="card-body">
                            <table class="data-table">
                                <thead>
                                    <tr>
                                        <th>Dataset</th>
                                        <th>Values</th>
                                    </tr>
                                </thead>
                                <tbody id="data-table-body">
                                    <!-- Table data will be populated here -->
                                </tbody>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </main>

    <script>
        function calculateStats() {
            const input = document.getElementById('ages-input').value.trim();

            if (!input) {
                showError('Please enter student ages to analyze.');
                return;
            }

            try {
                // Parse and validate input
                const ages = input.split(',').map(age => {
                    const num = parseInt(age.trim());
                    if (isNaN(num) || num <= 0 || num > 120) {
                        throw new Error('Invalid age value');
                    }
                    return num;
                });

                if (ages.length === 0) {
                    throw new Error('No valid ages found');
                }

                // Perform calculations
                const sortedAges = [...ages].sort((a, b) => a - b);
                const minAge = Math.min(...ages);
                const maxAge = Math.max(...ages);
                const median = calculateMedian(sortedAges);
                const average = ages.reduce((sum, age) => sum + age, 0) / ages.length;
                const range = maxAge - minAge;
                const minDistance = Math.abs(minAge - average);
                const maxDistance = Math.abs(maxAge - average);

                // Display results
                displayResults({
                    original: ages,
                    sorted: sortedAges,
                    min: minAge,
                    max: maxAge,
                    median: median,
                    average: average,
                    range: range,
                    minDistance: minDistance,
                    maxDistance: maxDistance,
                    count: ages.length
                });

            } catch (error) {
                showError('Invalid input format. Please ensure all values are valid ages between 1-120, separated by commas.');
            }
        }

        function calculateMedian(sortedArray) {
            const length = sortedArray.length;
            const middle = Math.floor(length / 2);

            if (length % 2 === 0) {
                return (sortedArray[middle - 1] + sortedArray[middle]) / 2;
            } else {
                return sortedArray[middle];
            }
        }

        function displayResults(stats) {
            const resultsDiv = document.getElementById('results');
            const statsGrid = document.getElementById('stats-grid');
            const comparisonContent = document.getElementById('comparison-content');
            const comparisonResult = document.getElementById('comparison-result');
            const dataTableBody = document.getElementById('data-table-body');

            // Populate stats grid
            statsGrid.innerHTML = `
                <div class="stat-card">
                    <div class="stat-header">
                        <div class="stat-title">Sample Size</div>
                        <div class="stat-icon">N</div>
                    </div>
                    <div class="stat-value">${stats.count}</div>
                    <div class="stat-description">Total number of students in dataset</div>
                </div>
            <div class="stat-card">
                <div class="stat-header">
                    <div class="stat-title">Minimum Age</div>
                    <div class="stat-icon">↓</div>
                </div>
                <div class="stat-value">${stats.min}</div>
                <div class="stat-description">Youngest student in the dataset</div>
            </div>

            <div class="stat-card">
                <div class="stat-header">
                    <div class="stat-title">Maximum Age</div>
                    <div class="stat-icon">↑</div>
                </div>
                <div class="stat-value">${stats.max}</div>
                <div class="stat-description">Oldest student in the dataset</div>
            </div>

            <div class="stat-card">
                <div class="stat-header">
                    <div class="stat-title">Median Age</div>
                    <div class="stat-icon">⌀</div>
                </div>
                <div class="stat-value">${stats.median}</div>
                <div class="stat-description">Middle value when sorted</div>
            </div>

            <div class="stat-card">
                <div class="stat-header">
                    <div class="stat-title">Average Age</div>
                    <div class="stat-icon">μ</div>
                </div>
                <div class="stat-value">${stats.average.toFixed(2)}</div>
                <div class="stat-description">Arithmetic mean of all ages</div>
            </div>

            <div class="stat-card">
                <div class="stat-header">
                    <div class="stat-title">Range</div>
                    <div class="stat-icon">R</div>
                </div>
                <div class="stat-value">${stats.range}</div>
                <div class="stat-description">Difference between max and min</div>
            </div>
        `;

        // Populate comparison
        comparisonContent.innerHTML = `
            <div class="comparison-item">
                <div class="comparison-label">|min - average|</div>
                <div class="comparison-value">${stats.minDistance.toFixed(2)}</div>
            </div>
            <div class="comparison-item">
                <div class="comparison-label">|max - average|</div>
                <div class="comparison-value">${stats.maxDistance.toFixed(2)}</div>
            </div>
```

```
            `;

            const comparisonText = stats.minDistance < stats.maxDistance
                ? 'Minimum age is closer to the average than maximum age'
                : stats.minDistance > stats.maxDistance
                ? 'Maximum age is closer to the average than minimum age'
                : 'Both minimum and maximum ages are equally distant from the average';

            comparisonResult.innerHTML = `
                <strong>Analysis Result:</strong><br>
                ${comparisonText}
            `;

            // Populate data table
            dataTableBody.innerHTML = `
                <tr>
                    <td><strong>Original Dataset</strong></td>
                    <td>[${stats.original.join(', ')}]</td>
                </tr>
                <tr>
                    <td><strong>Sorted Dataset</strong></td>
                    <td>[${stats.sorted.join(', ')}]</td>
                </tr>
            `;

            resultsDiv.style.display = 'block';
        }

        function showError(message) {
            const resultsDiv = document.getElementById('results');
            const statsGrid = document.getElementById('stats-grid');

            statsGrid.innerHTML = `<div class="error-message">${message}</div>`;
            resultsDiv.style.display = 'block';
        }

        function resetAnalysis() {
            document.getElementById('ages-input').value = '';
            document.getElementById('results').style.display = 'none';
        }

        // Auto-calculate on page load
        window.addEventListener('load', function() {
            calculateStats();
        });
    </script>
</body>
</html>
```

3. Object Extensibility and Sealing

a) Use the Object.preventExtensions method to prevent any further additions of properties to the student object.

b) Use the Object.isExtensible method to check if the student object is extensible. Store the result in a variable called extensibleStatus.

c) Create a new object called teacher with a 'subject' property set to 'Math'.

d) Use the Object.seal method to seal the teacher object, preventing any additions or deletions of properties.

e) Use the Object.isSealed method to check if the teacher object is sealed. Store the result in a variable called sealedStatus.

f) Print the extensibleStatus and sealedStatus to the console.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Object Extensibility & Sealing | JavaScript Pro</title>
    <style>
        :root {
            --primary-color: #0f172a;
            --secondary-color: #1e293b;
            --accent-color: #3b82f6;
            --success-color: #10b981;
            --warning-color: #f59e0b;
            --error-color: #ef4444;
            --neutral-50: #f8fafc;
            --neutral-100: #f1f5f9;
            --neutral-200: #e2e8f0;
            --neutral-300: #cbd5e1;
            --neutral-400: #94a3b8;
            --neutral-500: #64748b;
            --neutral-600: #475569;
            --neutral-700: #334155;
            --neutral-800: #1e293b;
            --neutral-900: #0f172a;
            --shadow-sm: 0 1px 2px 0 rgba(0, 0, 0, 0.05);
            --shadow-md: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.06);
            --shadow-lg: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);
            --shadow-xl: 0 20px 25px -5px rgba(0, 0, 0, 0.1), 0 10px 10px -5px rgba(0, 0, 0, 0.04);
        }

        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: 'Inter', -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, sans-serif;
            background: linear-gradient(135deg, var(--neutral-50) 0%, var(--neutral-100) 100%);
            min-height: 100vh;
            color: var(--neutral-800);
            line-height: 1.6;
        }

        .header {
            background: linear-gradient(135deg, var(--primary-color) 0%, var(--secondary-color) 100%);
            color: white;
            padding: 3rem 0;
            box-shadow: var(--shadow-lg);
            position: relative;
            overflow: hidden;
        }

        .header::before {
            content: '';
            position: absolute;
            top: 0;
            left: 0;
            right: 0;
            bottom: 0;
            background: url('data:image/svg+xml,<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 100 100"><defs><pattern id="grid" width="20" height="20" patternUnits="userSpa
            opacity: 0.4;
        }

        .header-content {
            max-width: 1200px;
            margin: 0 auto;
            padding: 0 2rem;
            text-align: center;
            position: relative;
            z-index: 1;
        }

        .header h1 {
            font-size: 3rem;
            font-weight: 800;
            margin-bottom: 1rem;
            letter-spacing: -0.025em;
            background: linear-gradient(135deg, #ffffff 0%, #cbd5e1 100%);
            -webkit-background-clip: text;
            -webkit-text-fill-color: transparent;
            background-clip: text;
        }

        .header p {
            font-size: 1.3rem;
            opacity: 0.9;
            font-weight: 300;
            max-width: 600px;
            margin: 0 auto;
        }

        .main-container {
            max-width: 1200px;
            margin: -2rem auto 0;
            padding: 0 2rem 4rem;
            position: relative;
            z-index: 2;
        }

        .demo-grid {
            display: grid;
            grid-template-columns: 1fr 1fr;
            gap: 2rem;
            margin-bottom: 2rem;
```

```
}

.demo-card {
    background: white;
    border-radius: 20px;
    box-shadow: var(--shadow-xl);
    border: 1px solid var(--neutral-200);
    overflow: hidden;
    transition: all 0.3s ease;
}

.demo-card:hover {
    transform: translateY(-4px);
    box-shadow: var(--shadow-xl), 0 0 0 1px var(--accent-color);
}

.card-header {
    background: linear-gradient(135deg, var(--neutral-50) 0%, var(--neutral-100) 100%);
    padding: 2rem;
    border-bottom: 1px solid var(--neutral-200);
    position: relative;
}

.card-header::after {
    content: '';
    position: absolute;
    bottom: 0;
    left: 2rem;
    right: 2rem;
    height: 3px;
    background: linear-gradient(90deg, var(--accent-color), var(--success-color));
    border-radius: 2px;
}

.card-title {
    font-size: 1.5rem;
    font-weight: 700;
    color: var(--neutral-800);
    margin-bottom: 0.5rem;
    display: flex;
    align-items: center;
    gap: 0.75rem;
}

.card-subtitle {
    color: var(--neutral-600);
    font-size: 1rem;
    font-weight: 500;
}

.card-body {
    padding: 2rem;
}

.object-info {
    background: var(--neutral-50);
    border: 1px solid var(--neutral-200);
    border-radius: 12px;
    padding: 1.5rem;
    margin-bottom: 1.5rem;
    font-family: 'Monaco', 'Menlo', monospace;
    font-size: 0.9rem;
}

.object-title {
    font-weight: 600;
    color: var(--neutral-700);
    margin-bottom: 0.75rem;
    font-family: 'Inter', sans-serif;
}

.object-content {
    color: var(--neutral-600);
    line-height: 1.5;
}

.status-badge {
    display: inline-flex;
    align-items: center;
    gap: 0.5rem;
    padding: 0.5rem 1rem;
    border-radius: 20px;
    font-size: 0.9rem;
    font-weight: 600;
    margin-top: 1rem;
}

.status-extensible {
    background: var(--success-color);
    color: white;
}

.status-not-extensible {
    background: var(--error-color);
    color: white;
}

.status-sealed {
    background: var(--warning-color);
    color: white;
}

.status-not-sealed {
    background: var(--success-color);
    color: white;
}

.action-button {
    width: 100%;
    padding: 1rem 2rem;
    background: linear-gradient(135deg, var(--accent-color) 0%, var(--secondary-color) 100%);
    color: white;
    border: none;
    border-radius: 12px;
    font-size: 1rem;
    font-weight: 600;
    cursor: pointer;
    transition: all 0.3s ease;
    text-transform: uppercase;
    letter-spacing: 0.05em;
    margin-bottom: 1rem;
}

.action-button:hover {
    transform: translateY(-2px);
    box-shadow: var(--shadow-lg);
}

.action-button:active {
    transform: translateY(0);
}

.console-output {
    background: var(--neutral-900);
    color: var(--neutral-100);
    border-radius: 12px;
    padding: 2rem;
    margin-top: 2rem;
    font-family: 'Monaco', 'Menlo', monospace;
    font-size: 0.9rem;
    line-height: 1.6;
    box-shadow: var(--shadow-lg);
    border: 1px solid var(--neutral-700);
}

.console-header {
    color: var(--accent-color);
    font-weight: 600;
    margin-bottom: 1rem;
    display: flex;
    align-items: center;
    gap: 0.5rem;
}

.console-line {
    margin-bottom: 0.5rem;
    padding: 0.25rem 0;
}

.console-command {
    color: var(--success-color);
}

.console-output-text {
    color: var(--neutral-300);
}

.console-result {
    color: var(--warning-color);
    font-weight: 600;
}

.results-grid {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 2rem;
    margin-top: 2rem;
}

.result-card {
    background: white;
    border-radius: 16px;
    padding: 2rem;
    box-shadow: var(--shadow-md);
    border: 1px solid var(--neutral-200);
    text-align: center;
}

.result-title {
    font-size: 1.1rem;
    font-weight: 600;
    color: var(--neutral-700);
    margin-bottom: 1rem;
}

.result-value {
    font-size: 2rem;
    font-weight: 800;
    margin-bottom: 0.5rem;
}

.result-description {
    color: var(--neutral-600);
    font-size: 0.9rem;
}

@media (max-width: 768px) {
    .header h1 {
        font-size: 2rem;
    }

    .header p {
        font-size: 1.1rem;
    }

    .main-container {
        padding: 0 1rem 2rem;
    }

    .demo-grid {
        grid-template-columns: 1fr;
    }

    .results-grid {
```

```
                grid-template-columns: 1fr;
            }

            .card-header,
            .card-body {
                padding: 1.5rem;
            }
        }
    </style>
</head>
<body>
    <header class="header">
        <div class="header-content">
            <h1>JavaScript Pro</h1>
            <p>Advanced Object Extensibility & Sealing Demonstration</p>
        </div>
    </header>

    <main class="main-container">
        <div class="demo-grid">
            <!-- Student Object Demo -->
            <div class="demo-card">
                <div class="card-header">
                    <h2 class="card-title">
                        🎓 Student Object
                    </h2>
                    <p class="card-subtitle">Object.preventExtensions() Demo</p>
                </div>
                <div class="card-body">
                    <div class="object-info">
                        <div class="object-title">Initial Student Object:</div>
                        <div class="object-content" id="student-object-display">
                            {<br>
                            &nbsp;&nbsp;name: "John Doe",<br>
                            &nbsp;&nbsp;age: 20,<br>
                            &nbsp;&nbsp;grade: "A"<br>
                            }
                        </div>
                    </div>

                    <button class="action-button" onclick="preventStudentExtensions()">
                        Prevent Extensions
                    </button>

                    <div id="student-status">
                        <div class="status-badge status-extensible">
                            ✓ Extensible
                        </div>
                    </div>
                </div>
            </div>

            <!-- Teacher Object Demo -->
            <div class="demo-card">
                <div class="card-header">
                    <h2 class="card-title">
                👨‍🏫 Teacher Object
            </h2>
            <p class="card-subtitle">Object.seal() Demo</p>
        </div>
        <div class="card-body">
            <div class="object-info">
                <div class="object-title">Initial Teacher Object:</div>
                <div class="object-content" id="teacher-object-display">
                    {<br>
                    &nbsp;&nbsp;subject: "Math"<br>
                    }
                </div>
            </div>

            <button class="action-button" onclick="sealTeacherObject()">
                Seal Object
            </button>

            <div id="teacher-status">
                <div class="status-badge status-not-sealed">
                    ✓ Not Sealed
                </div>
            </div>
        </div>
    </div>
</div>

<div class="results-grid">
    <div class="result-card">
        <div class="result-title">Extensible Status</div>
        <div class="result-value" id="extensible-result" style="color: var(--success-color);">true</div>
        <div class="result-description">Object.isExtensible(student)</div>
    </div>

    <div class="result-card">
        <div class="result-title">Sealed Status</div>
        <div class="result-value" id="sealed-result" style="color: var(--success-color);">false</div>
        <div class="result-description">Object.isSealed(teacher)</div>
    </div>
</div>

<div class="console-output">
    <div class="console-header">
        🖥️ Console Output
    </div>
    <div id="console-content">
        <div class="console-line">
            <span class="console-command">// Creating student object...</span>
        </div>
        <div class="console-line">
            <span class="console-output-text">Student object created with initial properties</span>
        </div>
        <div class="console-line">
            <span class="console-command">// Creating teacher object...</span>
        </div>
        <div class="console-line">
```

```
                <span class="console-output-text">Teacher object created with subject: "Math"</span>
            </div>
            <div class="console-line">
                <span class="console-result">Ready for demonstration!</span>
            </div>
        </div>
    </div>
</main>

<script>
    // Create student object
    const student = {
        name: "John Doe",
        age: 20,
        grade: "A"
    };

    // Create teacher object
    const teacher = {
        subject: "Math"
    };

    // Variables to store status
    let extensibleStatus = Object.isExtensible(student);
    let sealedStatus = Object.isSealed(teacher);

    // Update initial display
    updateDisplay();

    function preventStudentExtensions() {
        // Prevent extensions on student object
        Object.preventExtensions(student);

        // Check if extensible
        extensibleStatus = Object.isExtensible(student);

        // Update display
        updateStudentStatus();
        updateConsole("Student object extensions prevented");

        // Try to add a new property (this will fail silently)
        try {
            student.newProperty = "This won't work";
            updateConsole("Attempted to add new property: " + (student.newProperty || "undefined"));
        } catch (error) {
            updateConsole("Error adding property: " + error.message);
        }

        // Print to console
        console.log("extensibleStatus:", extensibleStatus);
        updateConsole("extensibleStatus: " + extensibleStatus);
    }

    function sealTeacherObject() {
        // Seal the teacher object
        Object.seal(teacher);
    // Check if sealed
    sealedStatus = Object.isSealed(teacher);

    // Update display
    updateTeacherStatus();
    updateConsole("Teacher object sealed");

    // Try to add a new property (this will fail silently)
    try {
        teacher.newProperty = "This won't work";
        updateConsole("Attempted to add new property: " + (teacher.newProperty || "undefined"));
    } catch (error) {
        updateConsole("Error adding property: " + error.message);
    }

    // Try to modify existing property (this should work)
    teacher.subject = "Physics";
    updateConsole("Modified existing property - subject: " + teacher.subject);

    // Print to console
    console.log("sealedStatus:", sealedStatus);
    updateConsole("sealedStatus: " + sealedStatus);
}

function updateStudentStatus() {
    const statusDiv = document.getElementById('student-status');
    if (extensibleStatus) {
        statusDiv.innerHTML = '<div class="status-badge status-extensible">✓ Extensible</div>';
    } else {
        statusDiv.innerHTML = '<div class="status-badge status-not-extensible">🔒 Not Extensible</div>';
    }

    document.getElementById('extensible-result').textContent = extensibleStatus;
    document.getElementById('extensible-result').style.color = extensibleStatus ? 'var(--success-color)' : 'var(--error-color)';
}

function updateTeacherStatus() {
    const statusDiv = document.getElementById('teacher-status');
    if (sealedStatus) {
        statusDiv.innerHTML = '<div class="status-badge status-sealed">🔒 Sealed</div>';
    } else {
        statusDiv.innerHTML = '<div class="status-badge status-not-sealed">✓ Not Sealed</div>';
    }

    document.getElementById('sealed-result').textContent = sealedStatus;
    document.getElementById('sealed-result').style.color = sealedStatus ? 'var(--warning-color)' : 'var(--success-color)';

    // Update teacher object display
    const teacherDisplay = document.getElementById('teacher-object-display');
    teacherDisplay.innerHTML = `
        {<br>
        &nbsp;&nbsp;subject: "${teacher.subject}"<br>
        }
    `;
}
```

```
        function updateDisplay() {
            updateStudentStatus();
            updateTeacherStatus();
        }

        function updateConsole(message) {
            const consoleContent = document.getElementById('console-content');
            const newLine = document.createElement('div');
            newLine.className = 'console-line';
            newLine.innerHTML = `<span class="console-output-text">${message}</span>`;
            consoleContent.appendChild(newLine);

            // Auto-scroll to bottom
            consoleContent.scrollTop = consoleContent.scrollHeight;
        }

        // Initial console output
        console.log("Initial extensibleStatus:", extensibleStatus);
        console.log("Initial sealedStatus:", sealedStatus);

        // Print status to console as required by assignment
        function printStatus() {
            console.log("extensibleStatus:", extensibleStatus);
            console.log("sealedStatus:", sealedStatus);
        }
    </script>
</body>
</html>
```

Assignment: Building a Student Management System Description: You are tasked with building a student management system using JavaScript. The system should allow you to perform various operations on a list of students, including adding, updating, deleting, and displaying student information. Requirements: Here is an initial array of students. Each student is represented as an object with the following properties: id, firstName, lastName, age, and grade. const students; Assignment Questions Full Stack Web Development Implement the following functions using pure JavaScript (without any external libraries or frameworks): a. Add a Student: Create a function to add a new student to the array. b. Update Student Information: Create a function to update a student's information based on their id. c. Delete a Student: Create a function to delete a student based on their id. d. List All Students: Create a function to display a list of all students. e. Find Students by Grade: Create a function to find all students who have a specific grade. f. Calculate Average Age: Create a function to calculate the average age of all students using array method.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Management System</title>
    <style>
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }
```

```
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
    min-height: 100vh;
    padding: 20px;
}

.container {
    max-width: 1200px;
    margin: 0 auto;
    background: white;
    border-radius: 20px;
    box-shadow: 0 20px 40px rgba(0,0,0,0.1);
    overflow: hidden;
}

.header {
    background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
    color: white;
    padding: 2rem;
    text-align: center;
}

.header h1 {
    font-size: 2.5rem;
    margin-bottom: 0.5rem;
    text-shadow: 2px 2px 4px rgba(0,0,0,0.3);
}

.header p {
    font-size: 1.1rem;
    opacity: 0.9;
}

.content {
    padding: 2rem;
}

.controls {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
    gap: 2rem;
    margin-bottom: 2rem;
}

.form-section {
    background: #f8f9fa;
    padding: 1.5rem;
    border-radius: 15px;
    box-shadow: 0 5px 15px rgba(0,0,0,0.08);
}

.form-section h3 {
    color: #333;
    margin-bottom: 1rem;
```

```css
    font-size: 1.3rem;
}

.form-group {
    margin-bottom: 1rem;
}

label {
    display: block;
    margin-bottom: 0.5rem;
    font-weight: 600;
    color: #555;
}

input[type="text"],
input[type="number"],
select {
    width: 100%;
    padding: 0.75rem;
    border: 2px solid #e9ecef;
    border-radius: 8px;
    font-size: 1rem;
    transition: border-color 0.3s ease;
}

input[type="text"]:focus,
input[type="number"]:focus,
select:focus {
    outline: none;
    border-color: #667eea;
    box-shadow: 0 0 0 3px rgba(102, 126, 234, 0.1);
}

.btn {
    background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
    color: white;
    border: none;
    padding: 0.75rem 1.5rem;
    border-radius: 8px;
    cursor: pointer;
    font-size: 1rem;
    font-weight: 600;
    transition: all 0.3s ease;
    text-transform: uppercase;
    letter-spacing: 0.5px;
}

.btn:hover {
    transform: translateY(-2px);
    box-shadow: 0 5px 15px rgba(102, 126, 234, 0.4);
}

.btn-secondary {
    background: linear-gradient(135deg, #6c757d 0%, #495057 100%);
}

.btn-danger {
```

```
.btn-danger {
    background: linear-gradient(135deg, #dc3545 0%, #c82333 100%);
}

.stats {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 1rem;
    margin-bottom: 2rem;
}

.stat-card {
    background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
    color: white;
    padding: 1.5rem;
    border-radius: 15px;
    text-align: center;
    box-shadow: 0 5px 15px rgba(0,0,0,0.1);
}

.stat-card h4 {
    font-size: 2rem;
    margin-bottom: 0.5rem;
}

.stat-card p {
    opacity: 0.9;
    font-size: 1.1rem;
}

.students-list {
    background: #f8f9fa;
    border-radius: 15px;
    padding: 1.5rem;
    box-shadow: 0 5px 15px rgba(0,0,0,0.08);
}

.students-list h3 {
    color: #333;
    margin-bottom: 1.5rem;
    font-size: 1.5rem;
}

.student-card {
    background: white;
    border-radius: 10px;
    padding: 1.5rem;
    margin-bottom: 1rem;
    box-shadow: 0 3px 10px rgba(0,0,0,0.1);
    transition: transform 0.3s ease;
}

.student-card:hover {
    transform: translateY(-3px);
    box-shadow: 0 5px 20px rgba(0,0,0,0.15);
}
```

```css
.student-info {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));
    gap: 1rem;
    align-items: center;
}

.student-name {
    font-size: 1.2rem;
    font-weight: 600;
    color: #333;
}

.student-details {
    color: #666;
    font-size: 0.9rem;
}

.grade-badge {
    display: inline-block;
    padding: 0.25rem 0.75rem;
    border-radius: 20px;
    font-weight: 600;
    font-size: 0.8rem;
    text-transform: uppercase;
}

.grade-a {
    background: #d4edda;
    color: #155724;
}

.grade-b {
    background: #d1ecf1;
    color: #0c5460;
}

.grade-c {
    background: #fff3cd;
    color: #856404;
}

.grade-d {
    background: #f8d7da;
    color: #721c24;
}

.student-actions {
    display: flex;
    gap: 0.5rem;
    margin-top: 1rem;
}

.btn-sm {
    padding: 0.4rem 0.8rem;
    font-size: 0.8rem;
```

```
}

.filter-section {
    display: flex;
    gap: 1rem;
    margin-bottom: 1rem;
    flex-wrap: wrap;
    align-items: center;
}

.alert {
    padding: 1rem;
    border-radius: 8px;
    margin-bottom: 1rem;
    font-weight: 600;
}

.alert-success {
    background: #d4edda;
    color: #155724;
    border: 1px solid #c3e6cb;
}

.alert-error {
    background: #f8d7da;
    color: #721c24;
    border: 1px solid #f5c6cb;
}

.no-students {
    text-align: center;
    padding: 2rem;
    color: #666;
    font-style: italic;
}

@media (max-width: 768px) {
    .container {
        margin: 0;
        border-radius: 0;
    }

    .header h1 {
        font-size: 2rem;
    }

    .controls {
        grid-template-columns: 1fr;
    }

    .stats {
        grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));
    }

    .student-info {
        grid-template-columns: 1fr;
```

```
                text-align: center;
            }

            .student-actions {
                justify-content: center;
            }

            .filter-section {
                flex-direction: column;
                align-items: stretch;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="header">
            <h1>🎓 Student Management System</h1>
            <p>Manage your students with ease and efficiency</p>
        </div>

        <div class="content">
            <div class="stats">
                <div class="stat-card">
                    <h4 id="totalStudents">0</h4>
                    <p>Total Students</p>
                </div>
                <div class="stat-card">
                    <h4 id="averageAge">0</h4>
                    <p>Average Age</p>
                </div>
                <div class="stat-card">
                    <h4 id="topGrade">N/A</h4>
                    <p>Most Common Grade</p>
                </div>
            </div>

            <div class="controls">
                <div class="form-section">
                    <h3>Add New Student</h3>
                    <div class="form-group">
                        <label for="firstName">First Name:</label>
                        <input type="text" id="firstName" placeholder="Enter first name">
                    </div>
                    <div class="form-group">
                        <label for="lastName">Last Name:</label>
                        <input type="text" id="lastName" placeholder="Enter last name">
                    </div>
                    <div class="form-group">
                        <label for="age">Age:</label>
                        <input type="number" id="age" placeholder="Enter age" min="1" max="100">
                    </div>
                    <div class="form-group">
                        <label for="grade">Grade:</label>
                        <select id="grade">
                            <option value="">Select Grade</option>
                            <option value="A">A</option>
                <option value="A">A</option>
                <option value="B">B</option>
                <option value="C">C</option>
                <option value="D">D</option>
            </select>
        </div>
        <button class="btn" onclick="addStudent()">Add Student</button>
    </div>

    <div class="form-section">
        <h3>Update Student</h3>
        <div class="form-group">
            <label for="updateId">Student ID:</label>
            <input type="number" id="updateId" placeholder="Enter student ID">
        </div>
        <div class="form-group">
            <label for="updateFirstName">First Name:</label>
            <input type="text" id="updateFirstName" placeholder="Enter first name">
        </div>
        <div class="form-group">
            <label for="updateLastName">Last Name:</label>
            <input type="text" id="updateLastName" placeholder="Enter last name">
        </div>
        <div class="form-group">
            <label for="updateAge">Age:</label>
            <input type="number" id="updateAge" placeholder="Enter age" min="1" max="100">
        </div>
        <div class="form-group">
            <label for="updateGrade">Grade:</label>
            <select id="updateGrade">
                <option value="">Select Grade</option>
                <option value="A">A</option>
                <option value="B">B</option>
                <option value="C">C</option>
                <option value="D">D</option>
            </select>
        </div>
        <button class="btn btn-secondary" onclick="updateStudent()">Update Student</button>
    </div>
</div>

<div id="alert" class="alert" style="display: none;"></div>

<div class="students-list">
    <h3>Student List</h3>
    <div class="filter-section">
        <label for="gradeFilter">Filter by Grade:</label>
        <select id="gradeFilter" onchange="filterStudents()">
            <option value="">All Grades</option>
            <option value="A">Grade A</option>
            <option value="B">Grade B</option>
            <option value="C">Grade C</option>
            <option value="D">Grade D</option>
        </select>
        <button class="btn btn-secondary" onclick="showAllStudents()">Show All</button>
    </div>
    <div id="studentsList"></div>
```

```
            </div>
        </div>
    </div>

    <script>
        // Initial array of students
        let students = [
            { id: 1, firstName: "John", lastName: "Doe", age: 20, grade: "A" },
            { id: 2, firstName: "Jane", lastName: "Smith", age: 22, grade: "B" },
            { id: 3, firstName: "Bob", lastName: "Johnson", age: 19, grade: "A" },
            { id: 4, firstName: "Alice", lastName: "Brown", age: 21, grade: "C" },
            { id: 5, firstName: "Charlie", lastName: "Davis", age: 23, grade: "B" }
        ];

        let nextId = 6;

        // Function to show alerts
        function showAlert(message, type = 'success') {
            const alert = document.getElementById('alert');
            alert.textContent = message;
            alert.className = `alert alert-${type}`;
            alert.style.display = 'block';

            setTimeout(() => {
                alert.style.display = 'none';
            }, 3000);
        }

        // Function to add a new student
        function addStudent() {
            const firstName = document.getElementById('firstName').value.trim();
            const lastName = document.getElementById('lastName').value.trim();
            const age = parseInt(document.getElementById('age').value);
            const grade = document.getElementById('grade').value;

            // Validation
            if (!firstName || !lastName || !age || !grade) {
                showAlert('Please fill in all fields', 'error');
                return;
            }

            if (age < 1 || age > 100) {
                showAlert('Age must be between 1 and 100', 'error');
                return;
            }

            // Create new student object
            const newStudent = {
                id: nextId++,
                firstName: firstName,
                lastName: lastName,
                age: age,
                grade: grade
            };

            // Add to students array
    students.push(newStudent);

    // Clear form
    document.getElementById('firstName').value = '';
    document.getElementById('lastName').value = '';
    document.getElementById('age').value = '';
    document.getElementById('grade').value = '';

    // Update display
    displayStudents();
    updateStats();
    showAlert(`Student ${firstName} ${lastName} added successfully!`);
}

// Function to update student information
function updateStudent() {
    const id = parseInt(document.getElementById('updateId').value);
    const firstName = document.getElementById('updateFirstName').value.trim();
    const lastName = document.getElementById('updateLastName').value.trim();
    const age = parseInt(document.getElementById('updateAge').value);
    const grade = document.getElementById('updateGrade').value;

    // Find student
    const studentIndex = students.findIndex(student => student.id === id);

    if (studentIndex === -1) {
        showAlert('Student not found', 'error');
        return;
    }

    // Update only non-empty fields
    if (firstName) students[studentIndex].firstName = firstName;
    if (lastName) students[studentIndex].lastName = lastName;
    if (age && age >= 1 && age <= 100) students[studentIndex].age = age;
    if (grade) students[studentIndex].grade = grade;

    // Clear form
    document.getElementById('updateId').value = '';
    document.getElementById('updateFirstName').value = '';
    document.getElementById('updateLastName').value = '';
    document.getElementById('updateAge').value = '';
    document.getElementById('updateGrade').value = '';

    // Update display
    displayStudents();
    updateStats();
    showAlert('Student updated successfully!');
}

// Function to delete a student
function deleteStudent(id) {
    const studentIndex = students.findIndex(student => student.id === id);

    if (studentIndex === -1) {
        showAlert('Student not found', 'error');
        return;
    }

    const student = students[studentIndex];
    if (confirm(`Are you sure you want to delete ${student.firstName} ${student.lastName}?`)) {
        students.splice(studentIndex, 1);
        displayStudents();
        updateStats();
        showAlert(`Student ${student.firstName} ${student.lastName} deleted successfully!`);
    }
}

// Function to display all students
function displayStudents(studentsToShow = students) {
    const studentsList = document.getElementById('studentsList');

    if (studentsToShow.length === 0) {
        studentsList.innerHTML = '<div class="no-students">No students found</div>';
        return;
    }

    studentsList.innerHTML = studentsToShow.map(student => `
        <div class="student-card">
            <div class="student-info">
                <div>
                    <div class="student-name">${student.firstName} ${student.lastName}</div>
                    <div class="student-details">ID: ${student.id}</div>
                </div>
                <div class="student-details">Age: ${student.age}</div>
                <div>
                    <span class="grade-badge grade-${student.grade.toLowerCase()}">Grade ${student.grade}</span>
                </div>
            </div>
            <div class="student-actions">
                <button class="btn btn-sm btn-danger" onclick="deleteStudent(${student.id})">Delete</button>
            </div>
        </div>
    `).join('');
}

// Function to find students by grade
function findStudentsByGrade(grade) {
    return students.filter(student => student.grade === grade);
}

// Function to filter students
function filterStudents() {
    const gradeFilter = document.getElementById('gradeFilter').value;

    if (gradeFilter === '') {
        displayStudents();
    } else {
        const filteredStudents = findStudentsByGrade(gradeFilter);
        displayStudents(filteredStudents);
    }
}

// Function to show all students
```

```
        // Function to show all students
        function showAllStudents() {
            document.getElementById('gradeFilter').value = '';
            displayStudents();
        }

        // Function to calculate average age
        function calculateAverageAge() {
            if (students.length === 0) return 0;

            const totalAge = students.reduce((sum, student) => sum + student.age, 0);
            return (totalAge / students.length).toFixed(1);
        }

        // Function to get most common grade
        function getMostCommonGrade() {
            if (students.length === 0) return 'N/A';

            const gradeCount = {};
            students.forEach(student => {
                gradeCount[student.grade] = (gradeCount[student.grade] || 0) + 1;
            });

            let maxCount = 0;
            let mostCommon = 'N/A';

            for (const grade in gradeCount) {
                if (gradeCount[grade] > maxCount) {
                    maxCount = gradeCount[grade];
                    mostCommon = grade;
                }
            }

            return mostCommon;
        }

        // Function to update statistics
        function updateStats() {
            document.getElementById('totalStudents').textContent = students.length;
            document.getElementById('averageAge').textContent = calculateAverageAge();
            document.getElementById('topGrade').textContent = getMostCommonGrade();
        }

        // Initialize the application
        function init() {
            displayStudents();
            updateStats();
        }

        // Start the application
        init();
    </script>
</body>
</html>
```