**Task 1: Basic HTML Structure**

**Objective: Create a basic HTML page with the essential structure.**

**Instructions:**

**Create an HTML document with the following structure:**

**A title in the head section.**

**A header with a page title.**

**A paragraph of text.**

**An unordered list with three items.**

**Task 2: Adding and Styling Links**

**Objective: Add links to the HTML page and style them with CSS.**

**Instructions:**

**Add a link to an external website in the paragraph.**

**Add a link to another section of the same page.**

**Use CSS to change the color of the links and remove the underline.**

**Task 3: Creating a Navigation Bar**

**Objective: Create a simple navigation bar with CSS.**

**Instructions:**

**Add a navigation bar with links to Home, About, and Contact pages.**

**Style the navigation bar to make it horizontal and visually appealing.**

**Task 4: Creating a Responsive Layout**

**Objective: Create a responsive layout using CSS Flexbox or Grid.**

**Instructions:**

**Create a layout with a header, main content area, and a footer.**

**Use Flexbox or Grid to ensure the layout is responsive and adjusts to different screen sizes.**

**Task 5: Styling Forms**

**Objective: Create and style a form with CSS.**

**Instructions:**

**Create a form with inputs for name, email, and a message.**

**Style the form to make it visually appealing and user-friendly.**

**Task 6: Advanced CSS - Animations**

**Objective: Add CSS animations to elements on the page.**

**Instructions:**

**Create a button with a hover animation.**

**Add a keyframe animation to an element (e.g., a div that changes color or moves across the screen).**

**Task 7: CSS Grid Layout**

**Objective: Create a complex layout using CSS Grid.**

**Instructions:**

**Create a layout with a header, sidebar, main content area, and footer.**

**Use CSS Grid to arrange these elements in a two-column layout with the sidebar and main content area taking up the majority of the width.**

**Ensure the layout is responsive, with the sidebar stacking below the main content area on smaller screens.**

**Task 8: CSS Variables and Custom Properties**

**Objective: Use CSS variables to create a themeable webpage.**

**Instructions:**

**Define CSS variables for primary colors, fonts, and spacing.**

**Apply these variables to various elements on the page.**

**Create a button that allows users to switch between two different themes by updating the CSS variables.**

**Task 9: Advanced Form Styling and Validation**

**Objective: Create a form with advanced styling and client-side validation.**

**Instructions:**

**Create a form with inputs for name, email, password, and a checkbox for terms and conditions.**

**Style the form to make it visually appealing.**

**Add client-side validation using HTML5 attributes and custom CSS for error states.**

**Task 10: Creating a Custom CSS Framework**

**Objective: Develop a small, reusable CSS framework.**

**Instructions:**

**Create a set of CSS classes for common elements such as buttons, forms, grids, and typography.**

**Ensure the classes are modular and can be easily reused in different projects.**

**Document the classes and provide examples of their usage.**

**Task 11: Responsive E-Commerce Product Grid with CSS Grid and Flexbox**

**Objective: Create a responsive e-commerce product grid with CSS Grid and Flexbox.**

**Instructions:**

**Create a grid layout for displaying products, including images, titles, prices, and "Add to Cart" buttons.**

**Ensure the layout is responsive, adjusting the number of columns based on screen size.**

**Add hover effects to the product cards to enhance user interaction.**

**Task 12: Building a CSS-Only Image Slider**

**Objective: Create an image slider using only HTML and CSS.**

**Instructions:**

**Create a slider with multiple images that automatically transitions every few seconds.**

**Use CSS animations and keyframes to achieve the sliding effect.**

**Ensure the slider is responsive and works well on different screen sizes.**

**Task 13: Advanced CSS Animation and Transitions**

**Objective: Create an interactive web page with advanced CSS animations and transitions.**

**Instructions:**

**Create an animated header that changes colors and scales up on hover.**

**Create a button that changes background color with a smooth transition on hover and click.**

**Add a modal dialog that slides in from the top when triggered by a button click, using CSS animations.**

**Task 14: Complex CSS Shapes and Clipping**

**Objective: Create complex shapes and effects using CSS clipping and masking.**

**Instructions:**

**Create a set of geometric shapes (e.g., hexagons, stars) using CSS.**

**Use the CSS clip-path property to create custom clipping paths for images.**

**Apply CSS masks to create advanced visual effects on images or elements.**

**Task 15: Advanced Parallax Scrolling Effect**

**Objective: Create a webpage with an advanced parallax scrolling effect using only CSS.**

**Instructions:**

**Create multiple sections with background images.**

**Apply a parallax scrolling effect to the background images using CSS.**

**Ensure the parallax effect works smoothly on different screen sizes and devices.**

**Task 16: Building a Custom Dropdown Menu with CSS and JavaScript**

**Objective: Create an advanced dropdown menu with CSS and JavaScript for interactions.**

**Instructions:**

**Create a multi-level dropdown menu that opens on hover and closes when clicking outside of it.**

**Style the dropdown menu with custom animations for opening and closing.**

**Ensure the menu is accessible with keyboard navigation and ARIA roles.**

**Task 17: Creating a Custom CSS Flexbox Grid System**

**Objective: Build a custom CSS grid system using Flexbox.**

**Instructions:**

**Design a responsive grid system with columns that can span different widths on various screen sizes.**

**Ensure the grid supports nesting, offsets, and order modifications.**

**Create a documentation page demonstrating the usage of the grid system with examples.**

**Task 18: Advanced CSS Grid Layout for a Magazine Page**

**Objective: Design a complex magazine-style layout using CSS Grid.**

**Instructions:**

**Create a magazine page layout with multiple sections such as a header, feature article, sidebars, and footers.**

**Use CSS Grid to define areas and ensure the layout adjusts elegantly for different screen sizes.**

**Include images, text blocks, and other elements to mimic a real magazine page.**

**Task 19: Implementing Dark Mode with CSS Variables and JavaScript**

**Objective: Create a toggleable dark mode feature using CSS variables and JavaScript.**

**Instructions:**

**Design a webpage with a toggle switch to switch between light and dark modes.**

**Use CSS variables to define colors for both light and dark themes.**

**Implement JavaScript to toggle the theme and persist the user's preference (optional).**

**Task 20: Creating a Responsive Navbar with CSS Grid and Flexbox**

**Objective: Build a responsive navigation bar using a combination of CSS Grid and Flexbox.**

**Instructions:**

**Design a navigation bar with a logo on the left and menu items on the right.**

**Use CSS Grid to structure the layout of the navigation bar.**

**Utilize Flexbox for aligning and spacing the menu items.**

**Ensure the navigation bar collapses into a hamburger menu on smaller screens.**

**Task 21: Building a CSS-Only Animated Progress Bar**

**Objective: Create an animated progress bar using CSS animations and keyframes.**

**Instructions:**

**Design a progress bar with customizable colors and animation effects.**

**Use CSS keyframes to animate the progress bar fill.**

**Ensure the progress bar is responsive and adjusts its width based on a percentage value.**

**Task 22: Advanced CSS Transitions for Image Gallery**

**Objective: Create a dynamic image gallery with advanced CSS transitions.**

**Instructions:**

**Design an image gallery with thumbnails and a main image display.**

**Use CSS transitions to smoothly swap images when clicking on thumbnails.**

**Implement a lightbox effect for viewing images in full size.**

**Task 23: Building a CSS-Only Tabbed Interface**

**Objective: Create a tabbed interface using only HTML and CSS for the structure and styling.**

**Instructions:**

**Design a tabbed interface with multiple tabs and content panels.**

**Use CSS for tab styling and transitions between active and inactive tabs.**

**Ensure clicking on tabs changes the displayed content panel without JavaScript.**