

EX NO:

DATE:

HTML AND CSS

AIM:

ALGORITHM:

1. Write an HTML program to demonstrate different types of lists. Include an unordered list (), an ordered list (), and a definition list (<dl>). Provide at least three items in each list with appropriate labels and descriptions in the definition list. Ensure the HTML document is structured and styled appropriately for clarity.

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Lists in HTML</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f5f5dc;
}
</style>
</head>
<body>
<h1>Types of Lists in HTML</h1>
<h2>Unordered List</h2>
<ul>
<li>Coffee</li>
<li>Tea</li>
<li>Milk</li>
</ul>
<h2>Ordered List</h2>
<ol>
<li>one</li>
<li>two</li>
<li>three</li>
</ol>
<h2>Definition List</h2>
<dl>
<dt>Coffee</dt>
<dd>A brewed drink prepared from roasted coffee beans
</dd>
<dt>Tea</dt>
<dd>A beverage made from the leaves of camelia sinensis plant</dd>
<dt>Milk</dt>
<dd>Milk is the primary source of food for humans</dd>
</dl>
</body>
</html>
```

OUTPUT:

Types of Lists in HTML

Unordered List

- Coffee
- Tea
- Milk

Ordered List

1. one
2. two
3. three

Definition List

Coffee	A brewed drink prepared from roasted coffee beans
Tea	A beverage made from the leaves of camelia sinensis plant
Milk	Milk is the primary source of food for humans

1. Write an HTML program to demonstrate the use of hyperlinks for navigation. a. Include hyperlinks that allow navigation within the same page, linking to specific sections using fragment identifiers. b. Include hyperlinks that navigate from one HTML page to another within the same directory. Ensure that each hyperlink is clearly labeled and that the

navigation within the page includes at least two sections with unique identifiers. Style the links to distinguish them from regular text for clarity.

PROGRAM:

Navigation.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Hyperlinks Navigation</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #e6e6fa;
}
a {
color: blue;
text-decoration: underline;
}
</style>
</head>
<body>
<h1>Hyperlinks Example</h1>
<a href="#section1">Go to Section 1</a> | <a href="#section2">Go to Section 2</a> |
<a href="book.html">Go to Books Website</a>
<h2 id="section1">Section 1</h2>
<p>This is Section 1.</p>
<h2 id="section2">Section 2</h2>
<p>This is Section 2.</p>
</body>
</html>
```

Book.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Books Website</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f0f8ff;
}
.book {
margin-bottom: 10px;
}
</style>
</head>
<body>
<h1>Goodreads</h1>
<div class="book">
<h2>The Great Gatsby</h2>
<p>by F. Scott Fitzgerald</p>
</div>

<div class="book">
<h2>To Kill a Mockingbird</h2>
<p>by Harper Lee</p>
</div>

<div class="book">
<h2>1984</h2>
<p>by George Orwell</p>
```

```
</div>
<a href="index.html">Back to Main Page</a>
</body>
</html>
```

OUTPUT:

Hyperlinks Example

[Go to Section 1](#) | [Go to Section 2](#) | [Go to Books Website](#)

Section 1

This is Section 1.

Section 2

This is Section 2.

Goodreads

The Great Gatsby

by F. Scott Fitzgerald

To Kill a Mockingbird

by Harper Lee

1984

by George Orwell

[Back to Main Page](#)

2. Create an HTML program to display a timetable using the <table> tag. a. Include headers for days of the week (Monday to Friday) and time slots (e.g., 9:00 AM to 5:00 PM). b. Populate the table cells with placeholders representing subjects or activities for each day and time slot.

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Timetable</title>
<style>
table {
width: 100%;
border-collapse: collapse;
background-color: #ffefd5;
}
table, th, td {
border: 1px solid black;
}
th, td {
padding: 10px;
text-align: center;
}
</style>
</head>
<body>
<h1>Weekly Timetable</h1>
<table>
<thead>
<tr>
<th>Time Slot</th>
<th>Monday</th>
<th>Tuesday</th>
```

```
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00 AM - 10:00 AM</td>
<td>Keyboard</td>
<td>Dance</td>
<td>Guitar</td>
<td>Music</td>
<td>Violin</td>
</tr>
<tr>
<td>10:00 AM - 11:00 AM</td>
<td>Music</td>
<td>Violin</td>
<td>Guitar</td>
<td>Keyboard</td>
<td>Dance</td>
</tr>
<tr>
<td>11:00 AM - 12:00 AM</td>
<td>Guitar</td>
<td>Dance</td>
<td>Keyboard</td>
<td>Music</td>
<td>Violin</td>
</tr>

</tbody>
</table>
</body>
</html>
```

OUTPUT:

Weekly Timetable

Time Slot	Monday	Tuesday	Wednesday	Thursday	Friday
9:00 AM - 10:00 AM	Keyboard	Dance	Guitar	Music	Violin
10:00 AM - 11:00 AM	Music	Violin	Guitar	Keyboard	Dance
11:00 AM - 12:00 AM	Guitar	Dance	Keyboard	Music	Violin

- 4.Implement a static home page using frames in HTML. a. Create a frame-based layout with a header frame, a navigation frame, a main content frame, and a footerframe. b. Include placeholder content in each frame:**
- The header frame should display the website title or logo.
 - The navigation frame should contain links to different sections or pages of the website.
 - The main content frame should display introductory text or images about the website.
 - The footer frame should display copyright information or contact details.
 - Ensure the frames are properly defined and styled for a clear and structured presentation of the home page.

PROGRAM:

Static.html

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

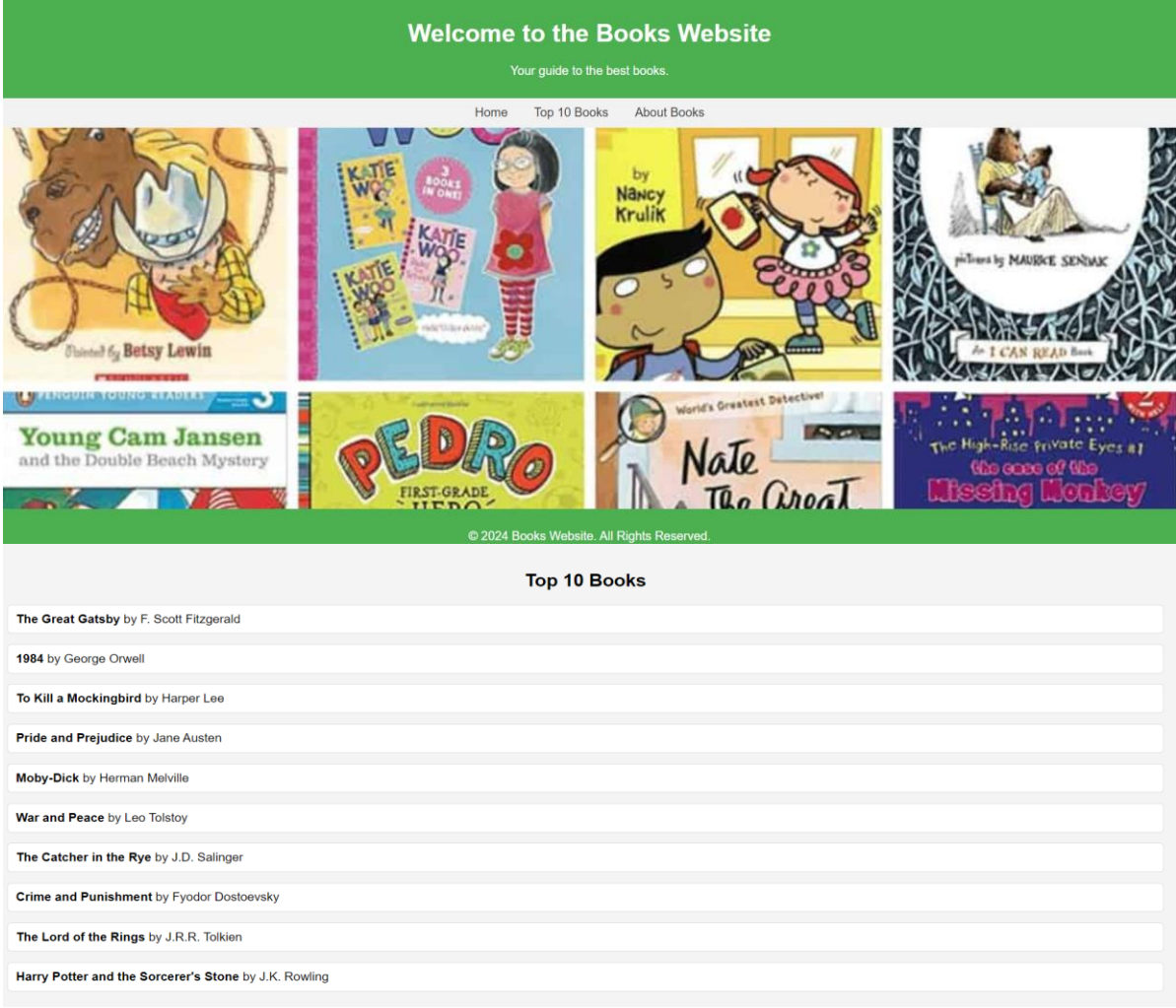
```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Books Website</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-image: url('kidsbook.jpeg');
background-size: cover;
color: #333;    }
header {
background-color: #4CAF50;
color: white;
text-align: center;
padding: 10px 0;
}
nav {
background-color: #f2f2f2;
padding: 10px;
text-align: center;
}
nav a {
margin: 0 15px;
text-decoration: none;
color: #333;
}
main {
padding: 20px;
}
footer {
background-color: #4CAF50;
color: white;
text-align: center;
padding: 10px 0;
position: absolute;
bottom: 0;
width: 100%; }
</style>
</head>
<body>
<header>
<h1>Welcome to the Books Website</h1>
<p>Your guide to the best books.</p>
</header>
<nav>
<a href="content.html">Home</a>
<a href="topbooks.html">Top 10 Books</a>
<a href="about.html">About Books</a>
</nav>
<footer>
<p>&copy; 2024 Books Website. All Rights Reserved.</p>
</footer>
</body>
</html>
```

Topbooks.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Top 10 Books</title>
<style>
body {
font-family: Arial, sans-serif;
padding: 20px;
```

```
background-color: #f4f4f4;
}
h2 {
text-align: center;
}
.book {
margin-bottom: 15px;
padding: 10px;
background-color: #ffffff;
border: 1px solid #ddd;
border-radius: 5px;
}
</style>
</head>
<body>
<h2>Top 10 Books</h2>
<div class="book"><b>The Great Gatsby</b> by F. Scott Fitzgerald</div>
<div class="book"><b>1984</b> by George Orwell</div>
<div class="book"><b>To Kill a Mockingbird</b> by Harper Lee</div>
<div class="book"><b>Pride and Prejudice</b> by Jane Austen</div>
<div class="book"><b>Moby-Dick</b> by Herman Melville</div>
<div class="book"><b>War and Peace</b> by Leo Tolstoy</div>
<div class="book"><b>The Catcher in the Rye</b> by J.D. Salinger</div>
<div class="book"><b>Crime and Punishment</b> by Fyodor Dostoevsky</div>
<div class="book"><b>The Lord of the Rings</b> by J.R.R. Tolkien</div>
<div class="book"><b>Harry Potter and the Sorcerer's Stone</b> by J.K. Rowling</div>
</body>
</html>
```

OUTPUT:



5.Develop a static registration form in HTML.a. Create a form with fields for a user to input their name, email address, password, date of birth, and country.b. Include appropriate labels (<label>) for each input field and use appropriate form

elements (<input>, <select>, <textarea>, etc.).c. Add a submit button (<input type="submit">) to allow users to submit their registration details.d. Ensure the form is styled appropriately for clarity and ease of use.

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Registration Form</title>
<style>
body {
background: url('flower.jpg') no-repeat center center fixed;
background-size: cover;
font-family: Arial, sans-serif;
color: black;
}
form {
width: 300px;
margin: 100px auto;
padding: 20px;
background: white;
box-shadow: 0px 0px 10px rgba(0,0,0,0.1);
border-radius: 10px;
}
h2 {
text-align: center;
}
label {
display: block;
margin: 10px 0 5px;
}
input[type="text"], input[type="email"], input[type="password"], input[type="date"], select {
width: 100%;
padding: 10px;
margin-bottom: 10px;
border: 1px solid #ccc;
border-radius: 5px;
box-sizing: border-box;
}
input[type="submit"] {
width: 100%;
background-color: green;
color: white;
border: none;
padding: 10px;
cursor: pointer;
border-radius: 5px;
}
input[type="submit"]:hover {
background-color: darkgreen;
}
</style>
</head>
<body>
<form>
<h2>Register Here</h2>
<label>Name:</label>
<input type="text" name="name" required>
<label>Email:</label>
<input type="email" name="email" required>
<label>Password:</label>
<input type="password" name="password" required>
<label>Date of Birth:</label>
<input type="date" name="dob" required>
```

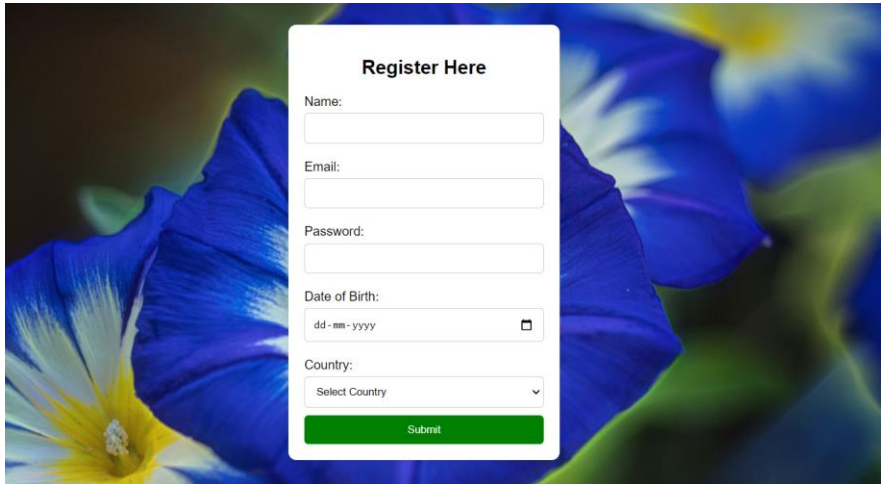


```

<label>Country:</label>
<select name="country" required>
<option value="">Select Country</option>
<option>USA</option>
<option>Canada</option>
<option>UK</option>
</select>
<input type="submit" value="Submit">
</form>
</body>
</html>

```

OUTPUT:



6. Write an HTML program to develop a static login page.

- Create a form with fields for username and password.
- Include appropriate labels (<label>) for each input field and use <input> elements with type="text" and type="password".
- Add a submit button (<input type="submit">) to allow users to submit their login credentials.
- Ensure the form is styled appropriately for clarity and ease of use.

PROGRAM:

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Login Page</title>
<style>
body {
font-family: Arial, sans-serif;
background: url('butterfly.png') no-repeat center center fixed;
background-size: cover;
margin: 0;
padding: 0;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
}
form {
background-color: rgba(255, 255, 255, 0.8);
padding: 20px;
border-radius: 8px;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
width: 300px;
}
h2 {
text-align: center;
margin-bottom: 20px;
}

```

```

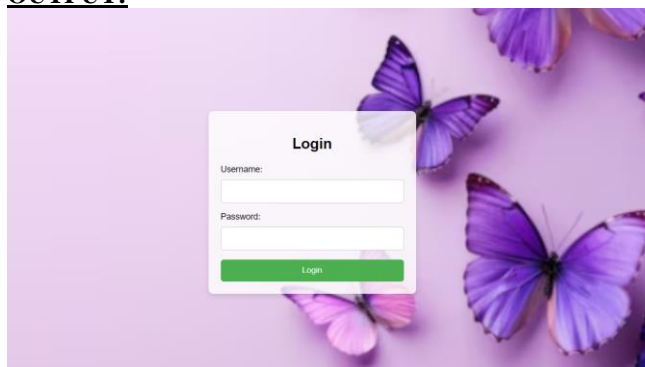
label {
display: block;
margin-bottom: 8px;
font-size: 14px;
}
input[type="text"], input[type="password"] {
width: 100%;
padding: 10px;
margin-bottom: 15px;
border: 1px solid #ccc;
border-radius: 5px;
box-sizing: border-box;
}
input[type="submit"] {
width: 100%;
padding: 10px;
background-color: #4CAF50;
color: white;
border: none;
border-radius: 5px;
cursor: pointer;
}
input[type="submit"]:hover {
background-color: #45a049;
}
</style>
</head>
<body>
<form>
<h2>Login</h2>
<label for="username">Username:</label>
<input type="text" id="username" name="username" required>

<label for="password">Password:</label>
<input type="password" id="password" name="password" required>

<input type="submit" value="Login">
</form>
</body>
</html>

```

OUTPUT:



7. Write an HTML program to develop a static web page for a catalog. a. Design a web page that displays a catalog of products or items. b. Each catalog item should include a product image, title, description, and price. c. Arrange the catalog items in a structured layout using <div>, , <h2>, <p>, and elements. d. Style the page to enhance visual appeal and readability, using appropriate CSS for layout, fonts, colors, and spacing.

PROGRAM:

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```
<title>Product Catalog</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-color: #f8f9fa;
}
header {
background-color: #343a40;
color: white;
padding: 20px;
text-align: left;
}
h1 {
margin: 0;
}
main {
padding: 20px;
}
.catalog {
display: flex;
flex-wrap: wrap;
justify-content: space-between;
}
.catalog-item {
background-color: white;
border: 1px solid #dee2e6;
border-radius: 5px;
box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
margin: 10px;
padding: 15px;
width: calc(48% - 20px);
text-align: center;
}
.catalog-item img {
max-width: 100px;
height: auto;
border-radius: 5px;
display: block;
margin: 0 auto;
}
h2 {
font-size: 1.5em;
margin: 10px 0;
}
p {
color: #6c757d;
}
.price {
font-size: 1.2em;
color: #28a745;
font-weight: bold;
}
</style>
</head>
<body>
<header>
<h1>Product Catalog</h1>
</header>
<main>
<div class="catalog">
<div class="catalog-item">

<h2>Pen</h2>
<p>A group of fancy pens</p>
```

```

<span class="price">$19.99</span>
</div>
<div class="catalog-item">

<h2>Ink Pens</h2>
<p>A group of ink pens</p>
<span class="price">$29.99</span>
</div>
</div>
</main>
</body>
</html>

```

OUTPUT:

Product Catalog



Pen

A group of fancy pens

\$19.99



Ink Pens

A group of ink pens

\$29.99

8. Write an HTML program to create a static web page for a catalog.

- Design a visually appealing web page that showcases a catalog of products.
- Each product entry should include an image, title, description, and price.
- Use appropriate HTML elements (`<div>`, ``, `<h2>`, `<p>`, ``) to structure and display each catalog item.
- Apply CSS styling to enhance the layout, typography, colors, and spacing for a professional look. Ensure the catalog is well-organized and the page is user-friendly with clear navigation and attractive presentation.

PROGRAM:

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Product Catalog</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-color: #f5f5f5;
}
header {
background-color: #007bff;
color: white;
padding: 20px;
text-align: center;
}
h1 {
margin: 0;
}
main {
padding: 20px;

```

```

}
.catalog {
display: flex;
flex-wrap: wrap;
justify-content: space-around;
}
.catalog-item {
background-color: white;
border: 1px solid #ddd;
border-radius: 8px;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
margin: 15px;
padding: 15px;
width: calc(30% - 30px); /* Three items per row */
text-align: center;
transition: transform 0.3s;
}
.catalog-item:hover {
transform: scale(1.05);
}

.catalog-item img {
max-width: 80%; /* Adjust image size */
height: auto;
border-radius: 5px;
display: block;
margin: 0 auto;
}
h2 {
font-size: 1.5em;
margin: 10px 0;
color: #333;
}
p {
color: #666;
font-size: 1em;
margin: 5px 0;
}
.price {
font-size: 1.2em;
color: #28a745;
font-weight: bold;
}
</style>
</head>
<body>
<header>
<h1>Catalog static webpage</h1>
</header>
<main>
<div class="catalog">
<div class="catalog-item">

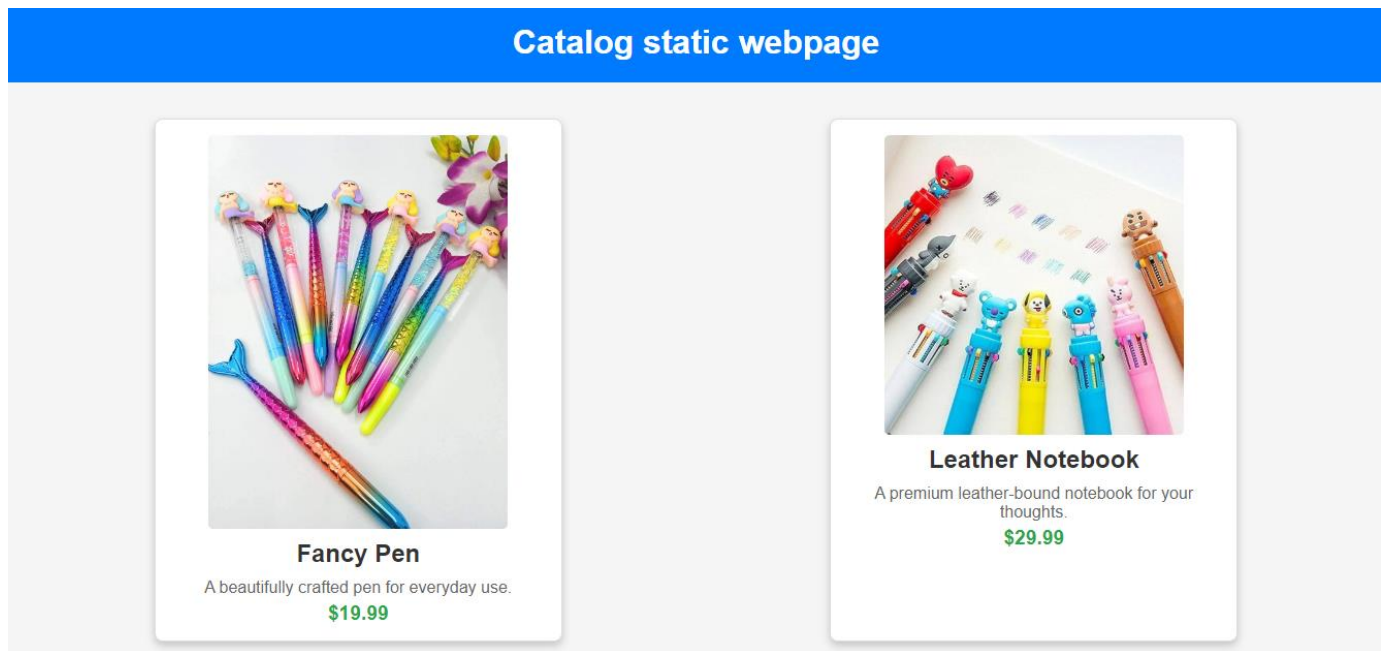
<h2>Fancy Pen</h2>
<p>A beautifully crafted pen for everyday use.</p>
<span class="price">$19.99</span>
</div>
<div class="catalog-item">

<h2>Leather Notebook</h2>
<p>A premium leather-bound notebook for your thoughts.</p>
<span class="price">$29.99</span>
</div>
</div>
</div>
</div>
</main>

```

</body>
</html>

OUTPUT:



9. Create an HTML document that demonstrates the use of cascading stylesheets by including examples of embedded stylesheets, external stylesheets, and inline styles. Ensure your HTML document includes: 1. An embedded stylesheet that defines styles for headings and paragraphs. 2. An external stylesheet that defines styles for a navigation bar and footer. 3. Inline styles for a specific element within the body. Include all necessary code for the HTML document, embedded stylesheet, and external stylesheet.

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Cascading Stylesheets Example</title>
<style>
body {
font-family: Arial, sans-serif;
margin: 0;
padding: 0;
background-color: #f5f5f5;
}
h1 {
color: #2c3e50;
text-align: center;
margin-top: 20px;
}
h2 {
color: #e74c3c;
}
p {
font-size: 1.1em;
color: #34495e;
margin: 10px 0;
}
nav {
background-color: #2980b9;
padding: 10px;
text-align: center;
}
```

```

nav ul {
list-style-type: none;
padding: 0;
}

nav ul li {
display: inline;
margin: 0 15px;
}
nav ul li a {
color: white;
text-decoration: none;
font-weight: bold;
}
</style>
</head>
<body>
<header>
<h1>CSS Demonstration</h1>
<nav>
<ul>
<li><a href="#home">Home</a></li>
<li><a href="#about">About</a></li>
<li><a href="#services">Services</a></li>
<li><a href="#contact">Contact</a></li>
</ul>
</nav>
</header>
<main>
<section>
<h2>Welcome to my webpage</h2>
<p>This is an example of a paragraph styled using internal CSS. It demonstrates how different CSS styles can be applied to various elements on the page.</p>
<p></p>
</section>
</main>
</body>
</html>

```

OUTPUT:



RESULT:

EX NO:

DATE:

JAVASCRIPT

AIM:

ALGORITHM:

1. Write a JavaScript to validate the fields of the login page

ii. Write a JavaScript to validate the fields of the Registration page (Should contain conditional logic, radio button, file upload, character limits, calendar adding, Drop down menus, etc)

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Login Page</title>
<script>
function validateLogin() {
const username = document.getElementById("username").value;
const password = document.getElementById("password").value;
const errorMessage = document.getElementById("error-message");
errorMessage.textContent = ""; // Clear previous messages

if (username === "") {
errorMessage.textContent = "Username cannot be empty.";
return false;
}
if (password === "") {
errorMessage.textContent = "Password cannot be empty.";
return false;
}
alert("Login successful!");
return true; // You can perform the actual login logic here
}
</script>
</head>
<body>
<h1>Login</h1>
<form onsubmit="return validateLogin()">
<label for="username">Username:</label>
<input type="text" id="username" required>
<br>
<label for="password">Password:</label>
<input type="password" id="password" required>
<br>
<button type="submit">Login</button>
<p id="error-message" style="color: red;"></p>
</form>
</body>
</html>
```

OUTPUT:

Login

Username:

Password:

This page says

Login successful!

OK

2. Simple calculator design

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Simple Calculator</title>
<style>
```

```
body {
display: flex;
flex-direction: column;
justify-content: center;
align-items: center;
height: 100vh;
background-color: #f5f5f5;
font-family: Arial, sans-serif;
}
h1 {
text-align: center;
margin-bottom: 20px;
}
.calculator {
border: 1px solid #ccc;
border-radius: 10px;
padding: 20px;
background-color: white;
box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
width: 300px;
}
#display {
width: 100%;
height: 40px;
font-size: 24px;
text-align: right;
margin-bottom: 20px;
padding: 5px;
border: 1px solid #ccc;
border-radius: 5px;
}
.button-grid {
display: grid;
grid-template-columns: repeat(4, 1fr);
gap: 10px;
}
button {
height: 50px;
font-size: 20px;
border: none;
border-radius: 5px;
background-color: #007bff;
color: white;
cursor: pointer;
transition: background-color 0.3s;
}
button:hover {
background-color: #0056b3;
}
button.operator {
background-color: #28a745;
}
button.operator:hover {
background-color: #218838;
}
button.clear {
background-color: #dc3545;
}
button.clear:hover {
background-color: #c82333;
}
</style>
</head>
<body>
<h1>Simple Calculator</h1>
<div class="calculator">
```

```

<input type="text" id="display" disabled>
<div class="button-grid">
<button onclick="clearDisplay()" class="clear">C</button>
<button onclick="appendToDisplay('7')">7</button>
<button onclick="appendToDisplay('8')">8</button>
<button onclick="appendToDisplay('9')">9</button>
<button onclick="appendToDisplay('/')" class="operator">/</button>

<button onclick="appendToDisplay('4')">4</button>
<button onclick="appendToDisplay('5')">5</button>
<button onclick="appendToDisplay('6')">6</button>
<button onclick="appendToDisplay('*')" class="operator">*</button>

<button onclick="appendToDisplay('1')">1</button>
<button onclick="appendToDisplay('2')">2</button>
<button onclick="appendToDisplay('3')">3</button>
<button onclick="appendToDisplay('-')" class="operator">-</button>

<button onclick="appendToDisplay('0')">0</button>
<button onclick="calculateResult()" class="operator">=</button>
<button onclick="appendToDisplay('+')" class="operator">+</button>
</div>
</div>
<script>
function appendToDisplay(value) {
const display = document.getElementById("display");
display.value += value;
}

function clearDisplay() {
document.getElementById("display").value = "";
}

function calculateResult() {
const display = document.getElementById("display");
try {
display.value = eval(display.value);
} catch (error) {
display.value = "Error";
}}
</script>
</body>
</html>

```

OUTPUT:

Simple Calculator



The image shows a simple calculator interface. It features a large rectangular display at the top. Below the display is a grid of buttons. The first row contains a red 'C' button (clear) followed by three blue buttons with digits '7', '8', and '9'. The second row contains a green '/' button followed by blue buttons '4', '5', and '6'. The third row contains a green '*' button followed by blue buttons '1', '2', and '3'. The fourth row contains a green '-' button, a blue '0' button, a green '=' button, and a green '+' button. The entire calculator is set against a light gray background.

3. Create a multi-step form with client-side validation at each step. Requirements: Divide the form into multiple steps (e.g., Personal Information, Account Details, Confirm Details). Each step should have its own set of fields:
Step 1: Name, Date of Birth (valid date format) **Step 2:** Username, Email, Password **Step 3:** Review and Confirm
Implement validation for each step: Allow navigation to the next step only if the current step's fields are valid. Provide a "Back" button to return to previous steps. Ensure the form is only submitted after all steps are completed and validated.

PROGRAM:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>User Registration and Resume Upload</title>
<style>
body {
font-family: Arial, sans-serif;
background-color: #f5f5f5;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
padding: 20px;
}
.form-container {
background-color: white;
border-radius: 10px;
box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
padding: 20px;
width: 400px;
}
.step, .login, .register {
display: none;
}
.active {
display: block;
}
input, select {
width: calc(100% - 12px);
padding: 10px;
margin: 10px 0;
border: 1px solid #ccc;
border-radius: 5px;
}
button {
padding: 10px 15px;
background-color: #007bff;
color: white;
border: none;
border-radius: 5px;
cursor: pointer;
margin-right: 10px;
}
button.back {
background-color: #6c757d;
}
button:disabled {
background-color: #ccc;
}
.link {
display: block;
margin-top: 10px;
text-align: center;
cursor: pointer;
color: #007bff;
}
</style>
</head>
<body>
<div class="form-container">
<div id="login" class="login active">
```

```

<h2>Login</h2>
<input type="text" id="loginUsername" placeholder="Username" required>
<input type="password" id="loginPassword" placeholder="Password" required>
<button onclick="login()">Login</button>
<div class="link" onclick="showRegistration()">Don't have an account? Register</div>
</div>
<div id="register" class="register">
<h2>Register</h2>
<input type="text" id="regName" placeholder="Name" required>
<input type="text" id="regUsername" placeholder="Username" required>
<input type="email" id="regEmail" placeholder="Email" required>
<input type="password" id="regPassword" placeholder="Password" required>
<button onclick="register()">Register</button>
<button class="back" onclick="showLogin()">Back</button>
</div>
<form id="resumeUploadForm">
<div class="step active" id="step1">
<h2>Personal Information</h2>
<input type="text" id="name" placeholder="Name" required>
<input type="date" id="dob" placeholder="Date of Birth" required>
<button type="button" onclick="nextStep()">Next</button>
</div>
<div class="step" id="step2">
<h2>Account Details</h2>
<input type="text" id="username" placeholder="Username" required>
<input type="email" id="email" placeholder="Email" required>
<input type="password" id="password" placeholder="Password" required>
<button type="button" class="back" onclick="prevStep()">Back</button>
<button type="button" onclick="nextStep()">Next</button>
</div>
<div class="step" id="step3">
<h2>Upload Resume</h2>
<input type="file" id="resume" accept=".pdf, .doc, .docx" required>
<select id="country" required>
<option value="">Select Country</option>
<option value="USA">USA</option>
<option value="Canada">Canada</option>
<option value="UK">UK</option>
<option value="Australia">Australia</option>
<!-- Add more countries as needed -->
</select>
<button type="button" class="back" onclick="prevStep()">Back</button>
<button type="button" onclick="nextStep()">Next</button>
</div>
<div class="step" id="step4">
<h2>Review and Confirm</h2>
<p><strong>Name:</strong> <span id="confirmName"></span></p>
<p><strong>Date of Birth:</strong> <span id="confirmDob"></span></p>
<p><strong>Username:</strong> <span id="confirmUsername"></span></p>
<p><strong>Email:</strong> <span id="confirmEmail"></span></p>
<p><strong>Country:</strong> <span id="confirmCountry"></span></p>
<button type="button" class="back" onclick="prevStep()">Back</button>
<button type="submit">Submit</button>
</div>
</form>
</div>
<script>
let currentStep = 0;
const steps = document.querySelectorAll(".step");

function showStep(step) {
  steps.forEach((s, index) => {
    s.classList.toggle("active", index === step);
  });
}

function nextStep() {

```

```

if (currentStep === 0) {
const name = document.getElementById("name").value;
const dob = document.getElementById("dob").value;
if (name && isValidDate(dob)) {
currentStep++;
showStep(currentStep);
} else {
alert("Please enter valid Name and Date of Birth.");
}
} else if (currentStep === 1) {
const username = document.getElementById("username").value;
const email = document.getElementById("email").value;
const password = document.getElementById("password").value;
if (username && validateEmail(email) && password) {
currentStep++;
showStep(currentStep);
} else {
alert("Please fill in all fields with valid data.");
}
} else if (currentStep === 2) {
const resume = document.getElementById("resume").files.length;
const country = document.getElementById("country").value;
if (resume > 0 && country) {
document.getElementById("confirmName").textContent = document.getElementById("name").value;
document.getElementById("confirmDob").textContent = document.getElementById("dob").value;
document.getElementById("confirmUsername").textContent = document.getElementById("username").value;
document.getElementById("confirmEmail").textContent = document.getElementById("email").value;
document.getElementById("confirmCountry").textContent = country;
currentStep++;
showStep(currentStep);
} else {
alert("Please upload a resume and select a country.");
}
}
}
function prevStep() {
currentStep--;
showStep(currentStep);
}
function validateEmail(email) {
const re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
return re.test(String(email).toLowerCase()); }
function isValidDate(dateString) {
const regex = /^\d{4}-\d{2}-\d{2}$/;
if (!dateString.match(regex)) return false;
const d = new Date(dateString);
return d && (d.getFullYear() === parseInt(dateString.split('-')[0]));
}
document.getElementById("resumeUploadForm").addEventListener("submit", function(event) {
event.preventDefault();
alert("Form submitted successfully!");
});
function login() {
const username = document.getElementById("loginUsername").value;
const password = document.getElementById("loginPassword").value;
if (username && password) {
alert("Login successful!");
// Transition to the multi-step form
showMultiStepForm();
} else {
alert("Please enter username and password.");
}
}
function register() {
const name = document.getElementById("regName").value;
const username = document.getElementById("regUsername").value;

```

```

const email = document.getElementById("regEmail").value;
const password = document.getElementById("regPassword").value;
if (name && username && validateEmail(email) && password) {
  alert("Registration successful!");
  showLogin();
} else {
  alert("Please fill in all fields with valid data.");
}
}

function showLogin() {
  document.getElementById("login").classList.add("active");
  document.getElementById("register").classList.remove("active");
  document.getElementById("resumeUploadForm").classList.remove("active");
}

function showRegistration() {
  document.getElementById("login").classList.remove("active");
  document.getElementById("register").classList.add("active");
  document.getElementById("resumeUploadForm").classList.remove("active");
}

function showMultiStepForm() {
  document.getElementById("login").classList.remove("active");
  document.getElementById("register").classList.remove("active");
  document.getElementById("resumeUploadForm").classList.add("active");
  currentStep = 0;
  showStep(currentStep);
}
</script>
</body>
</html>

```

OUTPUT:

Login


[Don't have an account? Register](#)

Personal Information

Account Details

Upload Resume

 sanjanah resume.pdf
 

3.Down counter design (with start, resume, pause, stop button)

PROGRAM:

```

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

```

```

<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Countdown Timer</title>
<style>
body {
font-family: Arial, sans-serif;
display: flex;
justify-content: center;
align-items: center;
height: 100vh;
background-color: #f5f5f5;
}
.timer-container {
text-align: center;
background-color: white;
border-radius: 10px;
box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
padding: 20px;
width: 300px;
}
h1 {
font-size: 48px;
margin: 20px 0;
}
.button {
padding: 10px 15px;
background-color: #007bff;
color: white;
border: none;
border-radius: 5px;
cursor: pointer;
margin: 5px;
}
.button:disabled {
background-color: #ccc;
}
</style>
</head>
<body>
<div class="timer-container">
<h1 id="timerDisplay">00:00:00</h1>
<input type="number" id="inputMinutes" placeholder="Minutes" min="1" />
<input type="number" id="inputSeconds" placeholder="Seconds" min="0" max="59" />
<div>
<button class="button" id="startButton">Start</button>
<button class="button" id="pauseButton" disabled>Pause</button>
<button class="button" id="resumeButton" disabled>Resume</button>
<button class="button" id="stopButton" disabled>Stop</button>
</div>
</div>
<script>
let countdown;
let remainingTime;
let isPaused = false;
const timerDisplay = document.getElementById("timerDisplay");
const startButton = document.getElementById("startButton");
const pauseButton = document.getElementById("pauseButton");
const resumeButton = document.getElementById("resumeButton");
const stopButton = document.getElementById("stopButton");
startButton.addEventListener("click", () => {
const minutes = parseInt(document.getElementById("inputMinutes").value);
const seconds = parseInt(document.getElementById("inputSeconds").value);
if (!isNaN(minutes) && !isNaN(seconds) && (minutes > 0 || seconds > 0)) {
remainingTime = (minutes * 60) + seconds;
startCountdown();
}
});

```



```

pauseButton.addEventListener("click", () => {
clearInterval(countdown);
isPaused = true;
toggleButtons();
});
resumeButton.addEventListener("click", () => {
startCountdown();
});
stopButton.addEventListener("click", () => {
clearInterval(countdown);
timerDisplay.textContent = "00:00:00";
document.getElementById("inputMinutes").value = "";
document.getElementById("inputSeconds").value = "";
isPaused = false;
toggleButtons();
});
function startCountdown() {
if (isPaused) {
isPaused = false;
} else {
toggleButtons();
}
countdown = setInterval(() => {
if (remainingTime <= 0) {
clearInterval(countdown);
timerDisplay.textContent = "Time's up!";
toggleButtons();
return;
}
remainingTime--;
updateDisplay();
}, 1000);}
function updateDisplay() {
const minutes = Math.floor(remainingTime / 60);
const seconds = remainingTime % 60;
timerDisplay.textContent = `${String(minutes).padStart(2, '0')}:${String(seconds).padStart(2, '0')}`;
}
function toggleButtons() {
startButton.disabled = !startButton.disabled;
pauseButton.disabled = !pauseButton.disabled;
resumeButton.disabled = !resumeButton.disabled;
stopButton.disabled = !stopButton.disabled;
}
</script>
</body>
</html>

```

OUTPUT:

00:00:00

Start

Pause

Resume

Stop

02:15

Start

Pause

Resume

Stop

RESULT:

EX NO:

DATE:

XML

AIM:

ALGORITHM:

1. Write an XML file which will display the Book information which includes the following: Title of the book, Author Name, ISBN number, Publisher name, Edition, Price, Write a Document Type Definition (DTD) to validate the above XML file. Display the XML file as follows. The contents should be displayed in a table. The header of the table should be in color GREY. And the Author names column should be displayed in one color and should be capitalized and in bold. Use your own colors for remaining columns. Use XML schemas XSL and CSS for the above purpose.

PROGRAM:

Books.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE books SYSTEM "books.dtd">
<?xml-stylesheet type="text/xsl" href="books.xsl"?>
<books>
  <book>
    <title>Effective Java</title>
    <author>Joshua Bloch</author>
    <isbn>978-0134685991</isbn>
    <publisher>Addison-Wesley</publisher>
    <edition>3rd</edition>
    <price>45.00</price>
  </book>
  <book>
    <title>Clean Code</title>
    <author>Robert C. Martin</author>
    <isbn>978-0132350884</isbn>
    <publisher>Prentice Hall</publisher>
    <edition>1st</edition>
    <price>37.95</price>
  </book>
</books>
```

books.dtd:

```
<!ELEMENT books (book+)>
<!ELEMENT book (title, author, isbn, publisher, edition, price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

books.xsl:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" doctype-public="XSLT-compat"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>Book Information</title>
        <link rel="stylesheet" type="text/css" href="styles.css"/>
      </head>
      <body>
        <h2>Book Information Table</h2>
        <table border="1">
          <thead style="background-color: grey; color: white;">
            <tr>
              <th>Title</th>
              <th>Author</th>
              <th>ISBN</th>
              <th>Publisher</th>
            </tr>
          </thead>
          <tbody>
            <xsl:for-each select="books/book">
              <tr>
                <td><xsl:value-of select="title"/></td>
                <td><xsl:value-of select="author"/></td>
                <td><xsl:value-of select="isbn"/></td>
                <td><xsl:value-of select="publisher"/></td>
              </tr>
            </xsl:for-each>
          </tbody>
        </table>
      </body>
    </html>
  </template>
</xsl:stylesheet>
```

```

        <th>Edition</th>
        <th>Price</th>
    </tr>
</thead>
<tbody>
    <xsl:for-each select="books/book">
        <tr>
            <td><xsl:value-of select="title"/></td>
            <td class="author"><xsl:value-of select="author"/></td>
            <td class="isbn"><xsl:value-of select="isbn"/></td>
            <td class="publisher"><xsl:value-of select="publisher"/></td>
            <td class="edition"><xsl:value-of select="edition"/></td>
            <td class="price"><xsl:value-of select="price"/></td>
        </tr>
    </xsl:for-each>
</tbody>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

styles.css:

```

body {
    font-family: Arial, sans-serif;
}
table {
    width: 80%;
    margin: auto;
    border-collapse: collapse;
}
th, td {
    padding: 8px;
    text-align: left;
}
.author {
    color: blue;
    font-weight: bold;
    text-transform: uppercase;
}
.isbn {
    color: darkred;
}
.publisher {
    color: darkgreen;
}
.edition {
    color: darkorange;
}
.price {
    color: darkviolet;
}

```

OUTPUT:

Book Information Table

Title	Author	ISBN	Publisher	Edition	Price
Effective Java	JOSHUA BLOCH	978-0134685991	Addison-Wesley	3rd	45.00
Clean Code	ROBERT C. MARTIN	978-0132350884	Prentice Hall	1st	37.95

2.XML with XSD Validation :Create an XML document representing a list of courses offered at a university. You will also create an XSD file to validate the structure of your XML document.

PROGRAM:

Courses.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="courses.xsl"?>
<university xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="courses.xsd">
  <course>
    <courseID>CS101</courseID>
    <courseName>Introduction to Computer Science</courseName>
    <credits>4</credits>
    <instructor>Dr. John Doe</instructor>
    <department>Computer Science</department>
    <semester>Fall</semester>
  </course>
  <course>
    <courseID>MATH201</courseID>
    <courseName>Calculus I</courseName>
    <credits>3</credits>
    <instructor>Prof. Jane Smith</instructor>
    <department>Mathematics</department>
    <semester>Spring</semester>
  </course>
  <course>
    <courseID>PHYS301</courseID>
    <courseName>Physics of Motion</courseName>
    <credits>4</credits>
    <instructor>Dr. Alan Brown</instructor>
    <department>Physics</department>
    <semester>Fall</semester>
  </course>
</university>
```

Courses.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="university">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="course" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="courseID" type="xs:string"/>
              <xs:element name="courseName" type="xs:string"/>
              <xs:element name="credits" type="xs:positiveInteger"/>
              <xs:element name="instructor" type="xs:string"/>
              <xs:element name="department" type="xs:string"/>
              <xs:element name="semester" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name="semesterType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Fall"/>
      <xs:enumeration value="Spring"/>
      <xs:enumeration value="Summer"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

OUTPUT:

CS101 Introduction to Computer Science 4Dr. John Doe Computer ScienceFall MATH201Calculus I 3Prof. Jane Smith MathematicsSpring PHYS301Physics of Motion 4Dr. Alan Brown PhysicsFall

3..XML with XSLT Transformation:Use the courses.xml file created in Exercise 1. Create an XSL file that transforms the XML data into an HTML table.

PROGRAM:

Courses.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html" encoding="UTF-8" />
  <xsl:template match="/university">
    <html>
      <head>
        <title>University Courses</title>
        <style>
          table {
            width: 100%;
            border-collapse: collapse;
          }
          th {
            background-color: grey;
            color: white;
            padding: 8px;
            text-align: left;
          }
          td {
            padding: 8px;
            text-align: left;
            border: 1px solid #ddd;
          }
          .courseID { color: blue; }
          .courseName { color: green; }
          .credits { color: red; }
          .instructor { color: purple; font-weight: bold; }
          .department { color: orange; }
          .semester { color: brown; }
        </style>
      </head>
      <body>
        <h2>List of Courses</h2>
        <table>
          <thead>
            <tr>
              <th>Course ID</th>
              <th>Course Name</th>
```

```

        <th>Credits</th>
        <th>Instructor</th>
        <th>Department</th>
        <th>Semester</th>
    </tr>
</thead>
<tbody>
    <xsl:for-each select="course">
        <tr>
            <td class="courseID"><xsl:value-of select="courseID"/></td>
            <td class="courseName"><xsl:value-of select="courseName"/></td>
            <td class="credits"><xsl:value-of select="credits"/></td>
            <td class="instructor"><xsl:value-of select="instructor"/></td>
            <td class="department"><xsl:value-of select="department"/></td>
            <td class="semester"><xsl:value-of select="semester"/></td>
        </tr>
    </xsl:for-each>
</tbody>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

OUTPUT:

List of Courses

Course ID	Course Name	Credits	Instructor	Department	Semester
CS101	Introduction to Computer Science	4	Dr. John Doe	Computer Science	Fall
MATH201	Calculus I	3	Prof. Jane Smith	Mathematics	Spring
PHYS301	Physics of Motion	4	Dr. Alan Brown	Physics	Fall

4.Advanced XML, XSD, and XSLT.Integration: Create an XML document representing an online store's product catalog. Validate the XML using XSD, and then create an XSL file to display the Catalog in a web-friendly format.

PROGRAM:

catalog.xml:

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE catalog SYSTEM "catalog.xsd">
<?xml-stylesheet type="text/xsl" href="catalog.xsl"?>
<catalog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="catalog.xsd">
    <product id="p001">
        <name>Smartphone</name>
        <description>Latest model with advanced features</description>
        <price currency="USD">699.99</price>
        <category>Electronics</category>
        <availability>In Stock</availability>
    </product>
    <product id="p002">
        <name>Laptop</name>
        <description>High-performance laptop for gaming and work</description>
        <price currency="USD">1299.99</price>
        <category>Computers</category>
        <availability>Out of Stock</availability>
    </product>
</catalog>

```

```
</product>
<product id="p003">
  <name>Wireless Headphones</name>
  <description>Noise-cancelling wireless headphones</description>
  <price currency="USD">199.99</price>
  <category>Accessories</category>
  <availability>In Stock</availability>
</product>
</catalog>
```

catalog.xsd:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="catalog">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="product" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="description" type="xs:string"/>
              <xs:element name="price">
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:decimal">
                      <xs:attribute name="currency" type="xs:string" use="required"/>
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
              <xs:element name="category" type="xs:string"/>
              <xs:element name="availability" type="xs:string"/>
            </xs:sequence>
            <xs:attribute name="id" type="xs:string" use="required"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

catalog.xsl:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head>
        <title>Online Store Product Catalog</title>
        <style>
          body { font-family: Arial, sans-serif; }
          table { width: 80%; margin: auto; border-collapse: collapse; }
          th, td { padding: 8px; text-align: left; border: 1px solid #ddd; }
          th { background-color: #f2f2f2; }
          tr:nth-child(even) { background-color: #f9f9f9; }
        </style>
      </head>
      <body>
        <h2 align="center">Online Store Product Catalog</h2>
        <table>
          <thead>
            <tr>
              <th>Product Name</th>
              <th>Description</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td>Product 1</td>
              <td>Description 1</td>
            </tr>
            <tr>
              <td>Product 2</td>
              <td>Description 2</td>
            </tr>
            <tr>
              <td>Product 3</td>
              <td>Description 3</td>
            </tr>
            <tr>
              <td>Product 4</td>
              <td>Description 4</td>
            </tr>
            <tr>
              <td>Product 5</td>
              <td>Description 5</td>
            </tr>
            <tr>
              <td>Product 6</td>
              <td>Description 6</td>
            </tr>
            <tr>
              <td>Product 7</td>
              <td>Description 7</td>
            </tr>
            <tr>
              <td>Product 8</td>
              <td>Description 8</td>
            </tr>
            <tr>
              <td>Product 9</td>
              <td>Description 9</td>
            </tr>
            <tr>
              <td>Product 10</td>
              <td>Description 10</td>
            </tr>
          </tbody>
        </table>
      </body>
    </html>
  </template>
</xsl:stylesheet>
```



```
<th>Price</th>
<th>Category</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<xsl:for-each select="catalog/product">
  <tr>
    <td><xsl:value-of select="name"/></td>
    <td><xsl:value-of select="description"/></td>
    <td><xsl:value-of select="price"/> <xsl:value-of select="price/@currency"/></td>
    <td><xsl:value-of select="category"/></td>
    <td><xsl:value-of select="availability"/></td>
  </tr>
</xsl:for-each>
</tbody>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

OUTPUT:

Online Store Product Catalog

Product Name	Description	Price	Category	Availability
Smartphone	Latest model with advanced features	699.99USD	Electronics	In Stock
Laptop	High-performance laptop for gaming and work	1299.99USD	Computers	Out of Stock
Wireless Headphones	Noise-cancelling wireless headphones	199.99USD	Accessories	In Stock

RESULT: